

Alma Mater Studiorum – Università di Bologna

DOTTORATO DI RICERCA

Fisica

Ciclo XIX

Settore scientifico disciplinare di afferenza: Fis/04

TITOLO TESI

Sistema di monitor e controllo della farm on-line
e studio del decadimento $B_s^0 \rightarrow J/\psi \phi$

Presentata dal dott. Daniele Gregori

Coordinatore Dottorato

prof. Fabio Ortolani

Relatore

prof. Domenico Galli

Correlatore

dott. Angelo Carbone

Esame finale anno 2008

Sistema di monitor e controllo della farm on-line e
studio del decadimento $B_s^0 \rightarrow J/\psi \phi$
a LHCb

dr. Daniele Gregori

14 marzo 2008

Ad Alteo.

Indice

Introduzione	1
1 L'esperimento LHCb	3
1.1 Produzione di mesoni B ad LHC	3
1.2 Il rivelatore LHCb	4
1.2.1 Il rivelatore di vertice	5
1.2.2 Il sistema di Pile-Up	8
1.2.3 Il magnete	9
1.2.4 La camera tracciante TT	11
1.2.5 Le stazioni traccianti T1, T2, T3	13
1.2.6 Il sistema dei RICH	16
1.2.7 I calorimetri	18
1.2.8 Le camere a muoni	24
1.2.9 Il sistema di trigger	24
2 Il sistema di monitor e di controllo	35
2.1 Il sistema di controllo dell'esperimento LHCb (ECS)	35
2.2 Il sistema di monitor e controllo della farm on-line (FMC)	39
2.3 Descrizione del sistema FMC	42
2.4 Il "middleware" DIM	43
2.5 L'Event Logger	44
2.5.1 L'estrazione dei messaggi dalla FIFO	45
2.5.2 Modalità di funzionamento "No-drop" e "Congestion-Proof"	47
2.5.3 Il filtro dei messaggi	47
2.5.4 La soppressione dei duplicati	48
2.5.5 Il "buffer" degli ultimi messaggi	49
2.5.6 Prestazioni dell'Event Logger	49
2.6 Il Power Manager	51
2.7 Il Task Manager	53
2.7.1 L'UTGID (User assigned Thread Group Identifier)	53
2.7.2 L'avvio di un processo	54
2.7.3 L'invio di un segnale a un processo	56
2.7.4 La terminazione di un processo	57
2.7.5 La lista dei processi in esecuzione	57
2.7.6 La reazione del Task Manager alla terminazione dei processi	58
2.7.7 Le prestazioni del Task Manager	60
2.8 Il Process Controller	61
2.8.1 Il riavvio rapido dei processi	62

2.8.2	Il controllo del riavvio	62
2.9	Il sistema di Monitor	63
2.9.1	I file system /proc e /sys	65
2.9.2	I parametri soggetti al monitor	65
2.10	Conclusioni	66
3	Il decadimento $B_s^0 \rightarrow J/\psi \phi$	67
3.1	Introduzione	67
3.2	Evoluzione temporale dei mesoni B	67
3.3	Decadimento istantaneo dei mesoni B	72
3.4	Il decadimento $B_s^0 \rightarrow J/\psi \phi$	74
3.4.1	La base trasversale	76
3.4.2	La distribuzione di probabilità del decadimento in un angolo	77
3.5	Studio di sensibilità della fase di mixing del B_s^0	80
3.6	Conclusioni	87
	Conclusioni	89

Introduzione

La tesi raccoglie il lavoro compiuto in collaborazione con il gruppo della Sezione dell' INFN di Bologna, per realizzare il sistema di configurazione e controllo della *farm* di *trigger* superiore (HLT, *High Level Trigger*) dell'esperimento LHCb. La tesi contiene inoltre un capitolo dedicato allo studio del decadimento del mesone $B_s \rightarrow J/\psi\phi$, particolarmente importante per la possibilità di determinare gli osservabili fisici che misurano la violazione della simmetria CP, dovuta al miscelamento particella-antiparticella.

Com'è noto, l'esperimento LHCb è uno degli esperimenti del *Large Hadron Collider* del CERN, progettato per compiere un ampio programma di misure di precisione delle proprietà e della dinamica del quark beauty, per mezzo dello studio dell'evoluzione temporale dei mesoni B neutri e mediante la ricerca di decadimenti rari dei mesoni B_d e B_s .

Benché LHC sia una copiosa sorgente di mesoni B, è necessario dotare il rivelatore LHCb di un sofisticato sistema di *trigger*, articolato in due livelli: il primo (denominato L0) è realizzato mediante circuiti elettronici, e ha la funzione di ridurre il rateo degli eventi dalla frequenza di interazione di LHC (40 MHz) a circa 1 MHz, mediante una selezione basata sul valore dell'impulso trasversale dei prodotti di decadimento.

Alla frequenza di 1 MHz opera il livello di *trigger* successivo (denominato HLT), realizzato mediante algoritmi *software* di selezione degli eventi, in esecuzione su di una *farm* di Personal Computer, interconnessi mediante una rete Gigabit Ethernet commutata.

Per sostenere il carico di lavoro previsto, la *farm* dev'essere costituita da qualche migliaio di CPU (circa 2000, secondo il progetto). La considerevole dimensione della *farm* di *trigger* può essere rapidamente valutata considerando che sulle CPU disponibili oggi, l'esecuzione dell'algoritmo di selezione del *trigger* richiede un tempo medio di esecuzione di qualche millisecondo, mentre il tempo disponibile in media per classificare un evento corrisponde all'inverso della frequenza massima di ingresso dei dati, che è pari a circa un microsecondo.

La *farm* di *trigger* deve funzionare con la massima efficienza, garantendo la piena affidabilità. Si è scelto di utilizzare il sistema operativo Linux appunto per ragioni di affidabilità e a garanzia di elevate prestazioni, non dimenticando peraltro il notevole vantaggio di poter disporre in questo modo di un sistema operativo *open source* totalmente gratuito.

I frammenti di evento prodotti dai diversi sotto-rivelatori di LHCb sono immessi alla frequenza di 1 MHz nella rete di *readout*, con un indirizzo di destinazione (indirizzo IP) pre-assegnato, corrispondente ogni volta ad un diverso nodo della *farm*. Il nodo destinatario opera di volta in volta l'*event building* e la successiva selezione dell'evento, segnalando — una volta completata la pro-

cedura — la sua disponibilità ad assumere un nuovo carico di lavoro. La distribuzione degli eventi ai nodi di calcolo, è arbitrata da un sistema di controllo: nella sua modalità di funzionamento più semplice, si prevede una distribuzione a rotazione (*round-robin*) degli eventi ai diversi nodi di calcolo.

La Sezione di Bologna dell'INFN ha assunto la responsabilità di sviluppare il sistema di controllo e di configurazione della *farm* (FMC, *Farm Monitoring and Control System*) e in questo ambito ho contribuito alla realizzazione dei programmi *software* responsabili della comunicazione con il sistema operativo dei nodi di calcolo per la raccolta dei parametri fondamentali per la determinazione del loro stato di funzionamento: occupazione della memoria RAM, stato dei dispositivi di rete, stato di occupazione dei dischi, temperature di lavoro, ecc. Ho inoltre contribuito alla realizzazione degli strumenti per la configurazione e controllo dei nodi di calcolo (accensione e spegnimento dei nodi, avvio e terminazione dei processi, raccolta dei messaggi inviati dai processi, ecc.).

Per completare il lavoro di tesi mi sono occupato della realizzazione dello studio di fattibilità della misura della fase di miscelamento dei mesoni B_s nel decadimento dei mesoni B_s nello stato finale costituito dalle particelle J/ψ e ϕ .

Dal punto di vista sperimentale il decadimento è di agevole rivelazione; costituisce una difficoltà il fatto che lo stato finale non abbia un valore di CP definito, con conseguente possibile diluizione del valore degli osservabili se non si separano le diverse componenti di CP. Il valore di CP dello stato finale non è definito perché lo stato finale — costituito da due particelle di spin uno — si deve trovare in uno stato di momento angolare totale nullo, con valori del momento orbitale nel sistema a riposo del mesone decaduto che producono stati di CP differenti. La separazione dei diversi stati di CP può essere realizzata su base statistica grazie all'osservazione della distribuzione angolare dei prodotti di decadimento.

Per questa tesi ho sviluppato il formalismo che permette di descrivere il decadimento in termini delle osservabili angolari. Ho quindi calcolato le distribuzioni di probabilità di decadimento e realizzato il codice per la simulazione dei processi di decadimento. Simulando serie ripetute di esperimenti, ciascuno corrispondente alla statistica di un anno di presa dati, ho determinato la precisione di misura raggiungibile nella determinazione della fase di miscelamento, controllando allo stesso tempo la consistenza del metodo di valutazione degli errori di misura mediante il metodo delle variabili *pull*.

La tesi è organizzata in tre capitoli. Nel primo sono fornite tutte le informazioni utili a inquadrare l'esperimento LHCb, con particolare rilievo ai sotto-rivelatori e alla descrizione del sistema di *trigger*. Nel secondo capitolo ho descritto il lavoro svolto per sviluppare il sistema di configurazione e controllo della *farm* di *trigger* HLT. Nel terzo e ultimo capitolo ho presentato il formalismo che permette di descrivere il decadimento e i risultati delle simulazioni. Seguono le conclusioni, in coda al lavoro.

Capitolo 1

L'esperimento LHCb

1.1 Produzione di mesoni B ad LHC

L'acceleratore LHC sarà un'intensa sorgente di adroni costituiti da quark beauty ed in particolare di mesoni B. I valori attesi per le sezioni d'urto di interazione e di produzione di beauty a 14 TeV di energia nel centro di massa sono elencate nella tabella (1.1).

Esse sono ricavate estrapolando i valori dai dati disponibili degli esperimenti UA1, CDF e D0 ¹ utilizzando il codice di simulazione PYTHIA [1].

Un parametro di funzionamento importante per LHCb è il numero medio di interazioni protone-protone che si verifica per incrocio dei fasci, dato da:

$$\langle N_{pp} \rangle = \frac{\mathcal{L}\sigma_{inel.}}{f_b} \quad (1.1)$$

dove \mathcal{L} è la luminosità istantanea, $\sigma_{inel.}$ è la sezione d'urto anelastica e f_b è la frequenza di incrocio dei fasci. Utilizzando il valore della sezione d'urto $\sigma_{inel.}$ riportato nella tabella (1.1), lavorando alla luminosità di $10^{34} \text{ cm}^{-2}\text{s}^{-1}$, alla frequenza di 40 MHz, sono attese mediamente 17.4 interazioni primarie protone-protone per incrocio dei fasci.

Nella regione di LHCb dove si vogliono avere con la massima probabilità eventi di interazione singola (molteplicità pari a uno), la luminosità istantanea di riferimento dev' essere di $2 \cdot 10^{32} \text{ cm}^{-2}\text{s}^{-1}$. In corrispondenza sono attese 0.34 interazioni per incrocio dei fasci. L'esperimento LHCb registrerà dunque

¹L'esatto valore della sezione d'urto all'energia di LHC non è ancora noto, il valore usato rappresenta un valore medio usato come riferimento per tutti gli esperimenti di LHC.

Tabella 1.1: Valori attesi per le sezioni d'urto all'energia di LHC.

$\sigma_{tot.}$	100 mb
$\sigma_{inel.}$	80 mb
$\sigma_{inel.} - \sigma_{diff.}$	55 mb
$\sigma_{b\bar{b}}$	500 μb
$\sigma_{c\bar{c}}$	1.5 μb

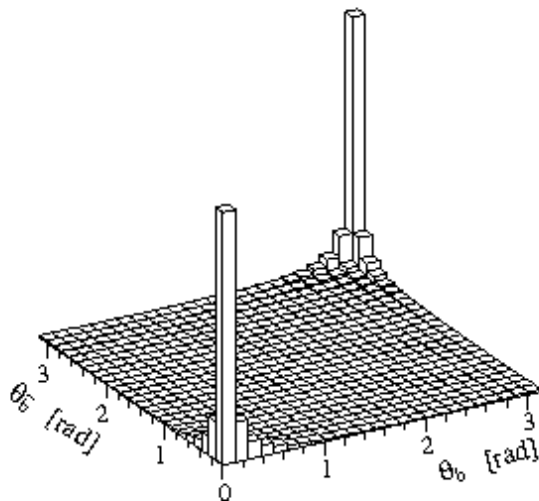


Figura 1.1: Angoli polari degli adroni b e \bar{b} prodotti con il generatore PYTHIA.

un'interazione nel 30% degli incroci dei fasci, con un rateo di interazioni effettivo di circa 12 MHz.

Il numero di coppie $b\bar{b}$ ottenute in un anno di presa dati (corrispondente a 10^7 s) è dell'ordine di 10^{12} . I due adroni che si producono nella successiva fase di adronizzazione sono emessi a piccoli angoli polari, come si vede in figura (1.1), ottenuta usando il codice di simulazione PYTHIA[1].

I mesoni B hanno un impulso medio di 80 GeV/c, cui corrisponde una lunghezza di decadimento media di circa 7 mm. La frazione di eventi b rispetto al totale delle interazioni è:

$$\frac{\sigma_{b\bar{b}}}{\sigma_{inel.}} = 0.6\% \quad (1.2)$$

Per questa ragione **un esperimento dedicato alla fisica del B dove essere dotato di un sistema di trigger in grado di selezionare gli eventi d'interesse.**

1.2 Il rivelatore LHCb

LHCb è uno spettrometro in avanti a un solo braccio, con accettanza geometrica compresa fra 10 mrad e 300 mrad sul piano orizzontale² e fra 10 mrad e 250 mrad sul piano verticale. Una rappresentazione del rivelatore è mostrata in figura (1.2).

Esso si estende su di una lunghezza di circa 20 m, è largo 10 m e alto 5 m. A partire dalla regione di interazione collocata a sinistra in figura (1.2), si susseguono i seguenti sotto-rivelatori:

- Il rivelatore di vertice.

²Questo è il piano di deflessione delle particelle cariche.

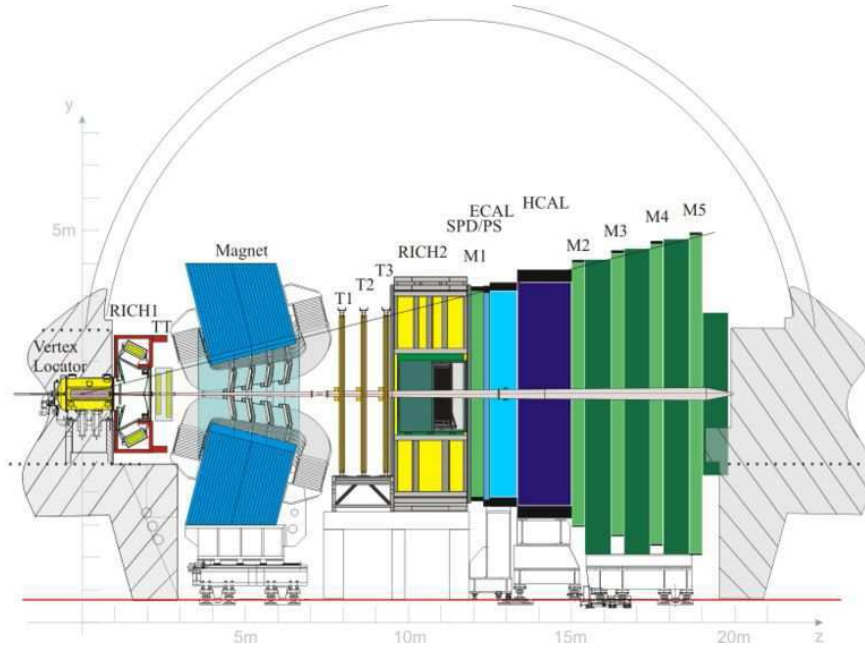


Figura 1.2: Schema dell'esperimento LHCb

- La camera TT del sistema di tracciamento, collocata prima del magnete.
- Il primo dei due rivelatori RICH (Ring Imaging Cherenkov) per la discriminazione delle particelle π/k di impulso minore di 10 GeV/c, emesse a grande angolo.
- Il magnete.
- Il secondo rivelatore RICH, dedicato all'identificazione di particelle di impulso compreso fra 10 GeV/c e 100 GeV/c.
- I calorimetri elettromagnetico e adronico.
- Le camere per la rivelazione dei muoni.

Segue la descrizione dei sotto-rivelatori.

1.2.1 Il rivelatore di vertice

Per poter realizzare il programma di fisica di LHCb, è fondamentale poter ricostruire i vertici secondari di decadimento dei mesoni B, con ottima risoluzione spaziale e temporale. La rivelazione di vertici secondari, distaccati dal vertice primario, costituisce un carattere peculiare del decadimento del mesone B, che, come anticipato, è atteso avere una distanza di volo media di circa 1 cm, a causa dell'impulso medio e della sua relativamente lunga vita media ($\tau \sim 10^{-12}$ s), vedi tabella (1.2.1).

Il rivelatore di vertice [2] [3] è lo strumento utilizzato allo scopo. La sua struttura è rappresentata in figura (1.3). È costituito da 21 stazioni di *micro-*

	massa [MeV]	vita media [$\times 10^{-12}$ s]
B_d^0	5279.4 ± 0.5	1.530 ± 0.009
B_s^0	5367.5 ± 0.6	1.466 ± 0.059

Tabella 1.2: Massa e vita media dei mesoni B neutri, ricavati dal “Particle data group” [4]

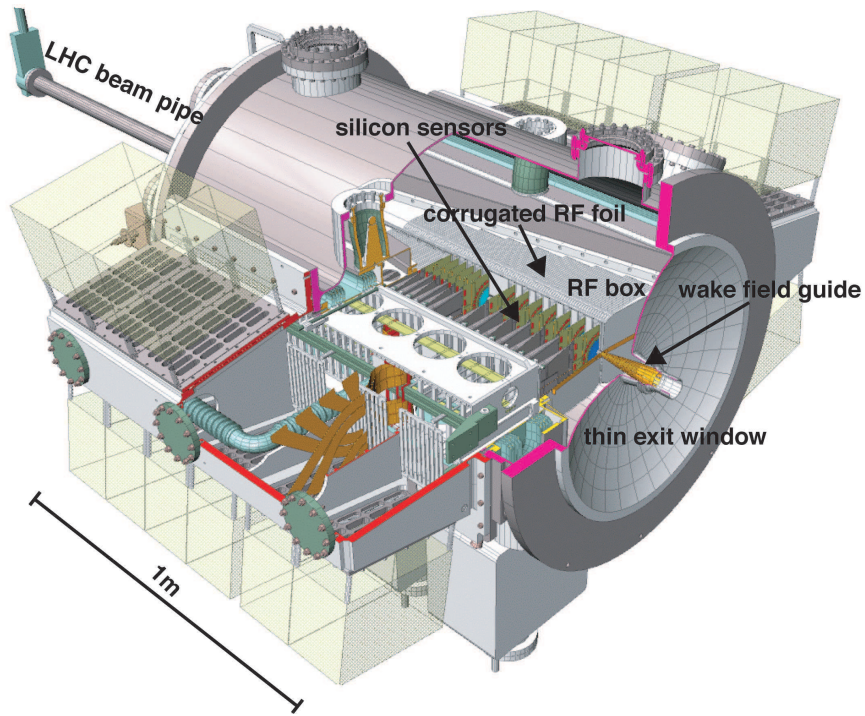


Figura 1.3: Sezione del rivelatore di vertice (VELO).

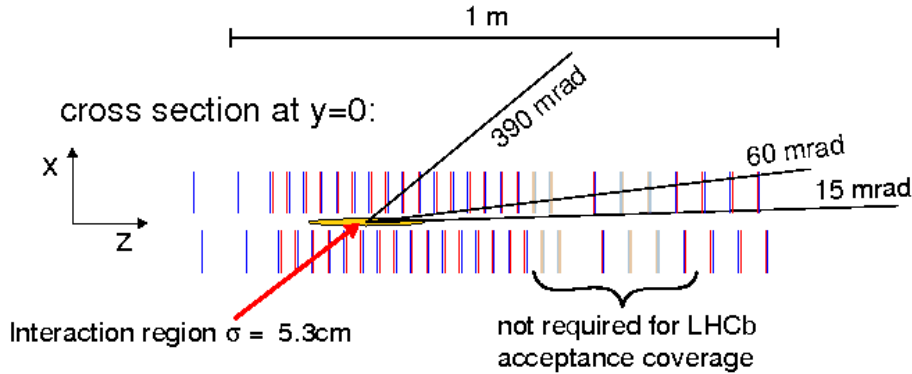


Figura 1.4: Disposizione delle stazioni traccianti del VELO. Le linee più chiare rappresentano le stazioni rimosse nel nuovo progetto del rivelatore. I due piani di sinistra disegnati in blu sono utilizzati come sistema di *Pile-Up*.

strip al silicio, ciascuna dello spessore di $220\text{ }\mu\text{m}$, posizionate perpendicolarmente alla linea di fascio che coprono una regione dell'estensione di circa 1 m , fra $z = -18\text{ cm}$ e $z = 80\text{ cm}$, come mostrato in figura (1.4).

Ciascuna stazione del rivelatore è costituita da due piani circolari di *strip* aventi un raggio di circa 4 cm , che consentono di misurare indipendentemente la distanza radiale R e la posizione azimutale φ delle *hit* prodotte nella propagazione delle particelle cariche, come mostrato in figura (1.5). I piani di rivelatore utilizzati per la misura della coordinata R sono suddivisi in 4 settori angolari di circa 45° e sono dotati di *strip* sagomate a forma di anello circolare. Il passo delle *micro-strip* è stato adattato [2] per rendere ottimale l'occupazione (*hit* per unità sensibile) dei canali in funzione della distanza radiale. La soluzione adottata prevede un andamento lineare crescente del passo delle *strip* in funzione della distanza radiale, variabile da un minimo di $40\text{ }\mu\text{m}$, nella regione più vicina al tubo del fascio, ad un massimo di $101.6\text{ }\mu\text{m}$ in quella più lontana.

I piani utilizzati per la misura della coordinata angolare φ sono suddivisi in due settori semi-circolari di raggio uguale a quello dei sensori R . Le *micro-strip*, hanno un passo variabile da $35.5\text{ }\mu\text{m}$ a $96.6\text{ }\mu\text{m}$, analogamente ai sensori R e sono orientate con angoli differenti passando dalle regioni più vicine al tubo del fascio a quelle più lontane.

Il rivelatore di vertice potrà essere estratto durante la fase di iniezione dei fasci per evitare danneggiamenti e portato a ridosso del fascio durante la presa dati, così da conseguire un'elevata accettazione di rivelazione nella regione a piccoli angoli polari. In condizioni di misura la parte sensibile si troverà alla distanza di 8.17 mm dalla linea di fascio e potrà essere ritirata fino a una distanza di 30 mm nella fase di iniezione.

Nella misura della posizione dei vertici primari ci si attende di poter raggiungere una precisione media, dell'ordine di $40\text{ }\mu\text{m}$ lungo z e di $10\text{ }\mu\text{m}$ sul piano trasversale, sia lungo x che lungo y . La risoluzione spaziale raggiungibile nella misura dei vertici secondari, naturalmente dipende dal numero di tracce utilizzabili per individuarlo e varia tra $100\text{ }\mu\text{m}$ e $300\text{ }\mu\text{m}$ a seconda del processo di decadimento specifico. La risoluzione in tempo proprio per i decadimenti dei mesoni B è attesa essere di 40 fs .

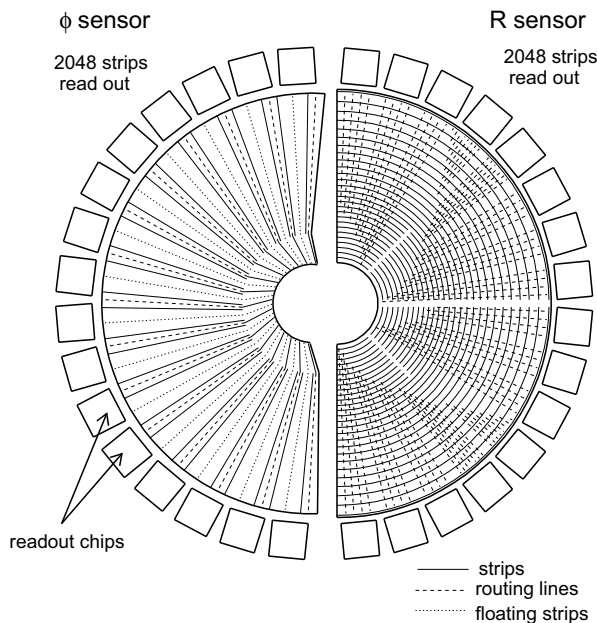


Figura 1.5: Rappresentazione della sagomatura dei semicerchi di *strip* del rivelatore di vertice, dedicati alla misura della coordinata angolare φ (a sinistra) e radiale R (a destra) delle hit prodotte dalle particelle cariche che attraversano il rivelatore.

1.2.2 Il sistema di Pile-Up

Per semplificare la ricostruzione degli eventi (almeno inizialmente), LHCb raccoglierà eventi con singola interazione primaria. Due stazioni del VELO (dotate solamente di sensori R) posizionate nella regione non coperta dal rivelatore, come mostrato in figura (1.4)), saranno utilizzate per determinare il numero di interazioni primarie che si verificano in ciascun incrocio dei fasci.

Le caratteristiche di dispersione del fascio nel piano trasversale (x, y) in prossimità della regione di interazione sono $\sigma_x = \sigma_y = \mathcal{O}(100 \mu\text{m})$, mentre lungo z ci si aspetta sia $\sigma_z \simeq 5 \text{ cm}$. La distribuzione dei vertici primari è dunque compresa in una regione $-15 \text{ cm} \leq z \leq 15 \text{ cm}$, mentre quella dei vertici di decadimento dei mesoni B è compresa in $-15 \text{ cm} \leq z \leq 20 \text{ cm}$.

Per stabilire quale sia la molteplicità delle interazioni primarie per ciascun incrocio dei fasci saranno utilizzate le misure delle distanza radiale R delle *hit* dalla linea di fascio che si ottengono in corrispondenza delle due stazioni collocate nelle posizioni di coordinata z pari a Z_a e Z_b .

Per mezzo della misura delle due distanze radiali R_a ed R_b , relative alle due stazioni, come mostrato in figura (1.6), si può determinare la misura della coordinata Z_{PV} del vertice primario. Infatti le *hit* prodotte da tracce che appartengono allo stesso vertice soddisfano la relazione:

$$Z_{PV} = \frac{kZ_a - Z_b}{k - 1} \quad (1.3)$$

dove $k = R_b/R_a$. L'equazione (1.3) è valida purché il vertice primario si trovi

con buona approssimazione sull'asse z . La distribuzione dei valori delle misure Z_{PV} che si ottiene considerando tutte le possibili combinazioni delle coordinate radiali R_a ed R_b secondo la relazione (1.3), presenta dei massimi locali in corrispondenza della posizione effettiva z dei vertici d'interazione. La distribuzione delle coordinate Z_{PV} permette di localizzare i vertici e di contare la molteplicità, come si può vedere dal grafico in basso della figura (1.6), dove è rappresentato ad esempio un evento nel quale si sono verificate due interazioni primarie.

Per inciso osserviamo che il sistema di *Pile-Up* potrebbe servire, oltre che per individuare gli eventi con interazione singola anche come monitor della luminosità³ [6]. La probabilità di avere n interazioni per incrocio dei fasci è descritta da una distribuzione poissoniana:

$$P(n) = \frac{\mu^n}{n!} e^{-\mu} \quad (1.4)$$

dove il numero medio di interazioni per incrocio di fascio μ , come abbiamo visto è proporzionale alla luminosità, essendo $\mu = \mathcal{L} \cdot \sigma_{det}$ dove σ_{det} è la sezione d'urto per l'efficienza del rivelatore diviso la frequenza d'incrocio dei fasci. In questo modo μ indica la luminosità relativa. Il *Pile-Up* è in grado di classificare gli incroci dei fasci in categorie contando il numero di vertici primari:

- Con zero interazioni N_0
- A singola interazione N_1
- A interazione multipla N_N , dove il pedice N indica il numero di vertici primari.

Grazie all'alta frequenza di incrocio dei fasci di LHC sarà possibile raccogliere in un tempo molto breve un'ampia statistica per ciascuna di queste tre categorie.

Ad esempio, alla luminosità di regime $2 \cdot 10^{32} \text{ cm}^{-2}\text{s}^{-1}$, si stima che per ciascuna delle tre classi le frazioni relative attese (ratei) saranno rispettivamente di 18 MHz, 9.3 MHz e 3 MHz. Si stima [6] una precisione sulla luminosità relativa dello 0.5% dopo 20 secondi di presa dati con 2652 incroci di fasci nel punto IP8.

1.2.3 Il magnete

Il magnete rappresentato in figura (1.7) è formato da 9 km di avvolgimenti di conduttori d'alluminio per un peso di 50 tonnellate e con ulteriori 120 mila tonnellate di acciaio. È in grado di fornire un campo magnetico integrato di 4 Tm (integrato su 10 m con 1.1 T di intensità massima) e dissiperà una potenza di circa 4 MW [7].

È posizionato in prossimità della regione d'interazione per contenere il più possibile le sue dimensioni. Il campo magnetico è diretto verticalmente in modo che il piano di deflessione sia quello (x, z) .

Esso copre un'accettanza angolare di 300 mrad sul piano di deflessione e di 250 mrad sul piano verticale. È un magnete caldo (non superconduttore) che permette di invertire il campo magnetico, per compensare eventuali asimmetrie destra-sinistra del rivelatore.

³Col termine monitorare la luminosità si intende una misura della variazione relativa di luminosità.

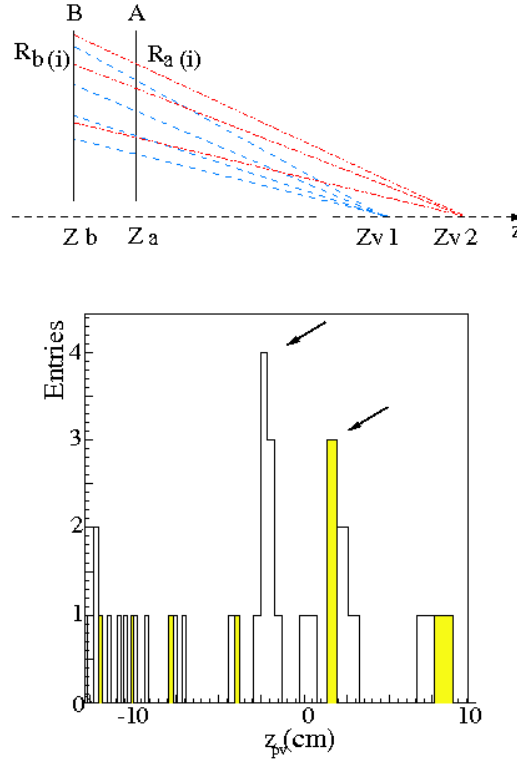


Figura 1.6: In alto: Rappresentazione schematica delle *hit* sui due piani di silicio nel sistema di *Pile-Up* delle tracce provenienti da due diversi vertici d'interazione. Le misure di R_a e R_b sono combinate in una matrice di coincidenze. In basso: istogramma per la variabile Z_{PV} . Per ciascuna combinazione delle misure R_a e R_b è calcolato il valore dell'equazione (1.3). I due picchi indicati con le frecce corrispondono ai due vertici d'interazione per questo particolare evento Monte Carlo. È attesa una efficienza sulla ricostruzione del vertice primario singolo del 95% e su quello doppio maggiore del 70% [5].



Figura 1.7: Foto del magnete montato all'interno della caverna di LHCb.

1.2.4 La camera tracciante TT

La camera tracciante denominata TT, mostrata in figura (1.8), è situata a 162 cm dal punto di interazione e copre una superficie di $157.2 \times 132.4 \text{ cm}^2$ corrispondente all'intera accettazione del rivelatore LHCb.

Questo dispositivo di rivelazione è formato da quattro piani di *strip* al silicio, posizionati fra VELO e magnete. I piani sono raggruppati a coppie, chiamate stazioni TTa e TTb, come mostrato in figura (1.8). Le *strip* del primo e quarto piano sono disposte parallelamente all'asse y mentre quelle del secondo e terzo piano sono ruotate di un angolo stereo di $+5^\circ$ e -5° rispettivamente, vedi figura (1.9), per permettere la misura della posizione delle *hit* sul piano (x, y) . Per tutte le *strip* il passo è $180 \mu\text{m}$. Tutti i piani sono racchiusi in un unico volume comune, che ha la funzione di mantenere la temperatura $< 5^\circ\text{C}$ e di isolarli otticamente e termicamente dall'ambiente esterno.

La funzione della stazione TT è duplice. Essa fornirà informazioni utili al tracciamento in fase di ricostruzione degli eventi acquisiti; inoltre assieme alle informazioni prodotte dal VELO, sarà impiegata per la misura dell'impulso trasversale delle tracce cariche nel *trigger* L1.

Per realizzare una selezione efficace si necessita della misura dell'impulso trasversale. Infatti, gli studi volti all'ottimizzazione degli algoritmi di selezione del *trigger* hanno mostrato che tracce di fondo, dovute alla diffrazione coulombiana multipla, apparivano dotate di un elevato valore del parametro d'impatto benché avessero impulsi trasversali relativamente piccoli.

Per tale ragione il sistema tracciante TT è posizionato tra il RICH1 e il magnete, in una regione dove è presente un campo magnetico residuo in grado di deflettere le particelle in modo da consentire una misura dell'impulso con precisione media del $20 \div 30\%$.

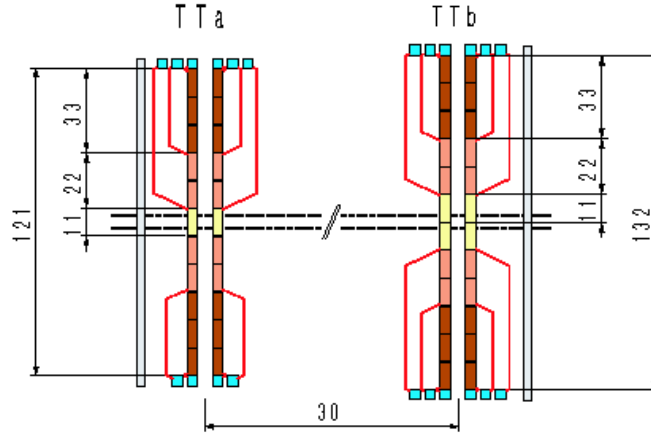


Figura 1.8: Vista schematica del sistema TT. I piani sono disposti a coppie (TTa e TTb) distanziate di 30 cm.

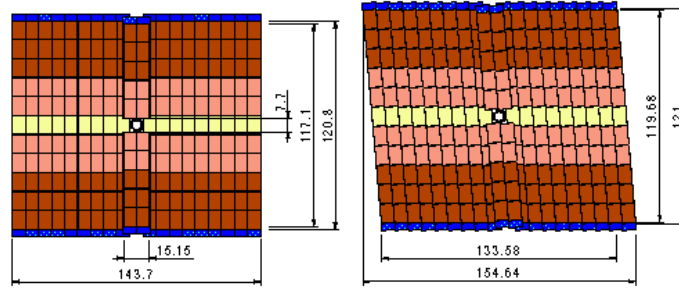


Figura 1.9: Vista schematica dei piani traccianti del sistema TT, interamente realizzati in *micro-strip* al silicio. A sinistra: piano tracciante con *strip* parallele all'asse y . A destra: piano tracciante con *strip* ruotate di 5° rispetto all'asse y . La camera TT è costituita da quattro piani di *strip* sovrapposti. Le *strip* del primo e del quarto piano sono posizionate parallelamente all'asse y mentre quelle del secondo e terzo piano sono ruotate di un angolo stereo di $+5^\circ$ e -5° rispettivamente.

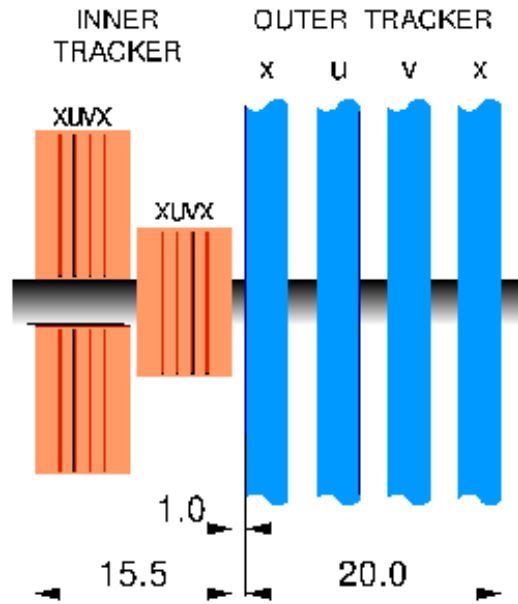


Figura 1.10: Vista schematica dei piani traccianti di una delle stazioni di tracciamento T. Le dimensioni sono in cm. E' possibile distinguere tra il sistema di tracciamento interno (IT) di colore rosa a sinistra e quello esterno (OT) di colore blu a destra.

1.2.5 Le stazioni traccianti T1, T2, T3

La funzione principale delle stazioni traccianti (T1, T2, T3), rappresentate schematicamente in figura (1.10) e situate oltre il magnete e prima del RICH2, è di misurare le coordinate prodotte dalle *hit* delle particelle cariche, per ricostruire le tracce lunghe⁴ in modo da misurare il momento delle particelle.

Le camere traccianti sono importanti anche perché forniscono la direzione delle tracce usate per la ricostruzione degli anelli Cherenkov nel rivelatore RICH. La misura delle tracce nelle stazioni T sono anche usate per agevolare la ricostruzione nei calorimetri e nelle camere a muoni.

Ogni stazione tracciante è composta da due parti, con diversa dimensione delle celle sensibili: la parte interna (*Inner Tracker*) e la parte esterna (*Outer Tracker*). Questa divisione è dovuta alla diversa densità di particelle al variare dell'angolo polare; si usano quindi due diversi sotto-rivelatori con diversa granularità.

È attesa una grande molteplicità di tracce a piccoli angoli polari, così da richiedere l'uso di sensori a micro *strip* al Silicio per il tracciamento interno. A grandi angoli polari, lontano dalla zona del fascio, la densità di particelle sarà moderata; questa regione sarà coperta dal tracciamento esterno usando camere a deriva.

⁴Come mostreremo in seguito, nella figura (1.14), le tracce lunghe sono quelle che attraversano il rivelatore nella sua interezza.

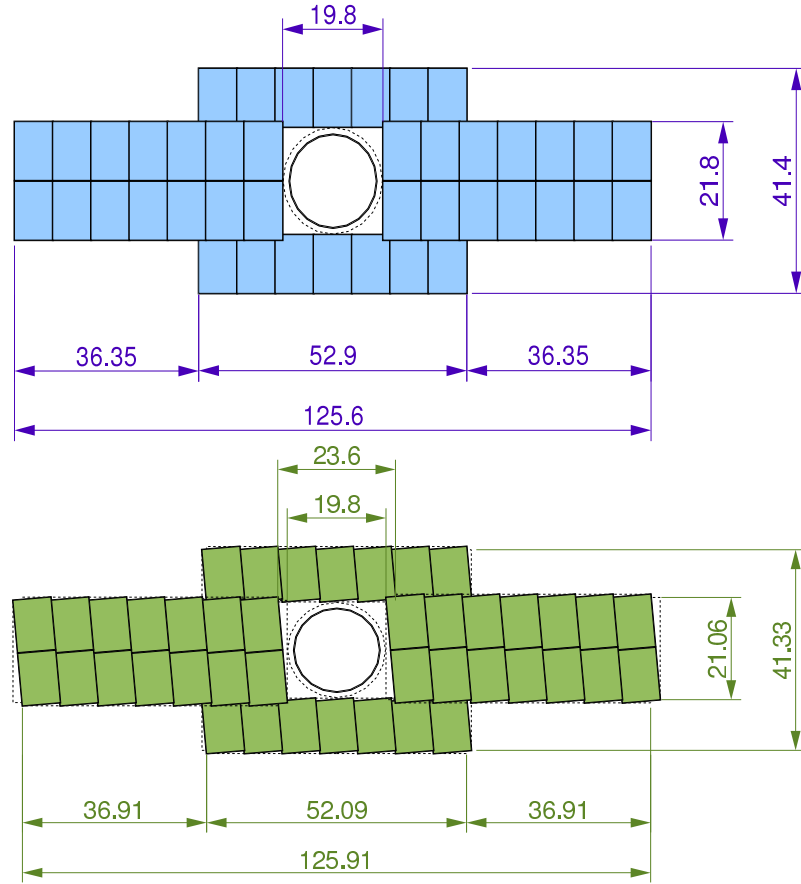


Figura 1.11: Schema dei sensori dell' IT nel piano (x,y). In alto: Camera di tipo x , con strip parallele all'asse y . In basso: Camere di tipo $u v$ con *strip* ruotata di un angolo stereo di 5° rispetto all' asse y .

Il tracciamento nella regione interna

La parte interna delle stazioni T1, T2, T3, quella vicino alla *beam pipe*, dove è maggiore il flusso di particelle, forma il sistema di tracciamento interno (IT) [8]. Ogni stazione consiste di quattro rivelatori indipendenti a forma rettangolare come schematizzati in figura (1.10). Ogni IT è composto da quattro camere indipendenti di sensori di *strip* al silicio larghi 78 mm e lunghi 110 mm. Lo spessore delle *strip* è di 198 mm. I sensori sono montati in una struttura a gradini [9] su una impalcatura che permette una piccola sovrapposizione tra sensori vicini. Sette gradini formano un piano di sensori. E ciascuna delle camere indipendenti contiene quattro piani, sovrapposti con orientazione alternativamente verticale e stereo $\pm 5^\circ$ come si vede in figura (1.11). La superficie totale coperta dall' IT è circa 120 cm^2 . La risoluzione spaziale sulla singola *hit* è dell'ordine di $70 \mu\text{m}$.

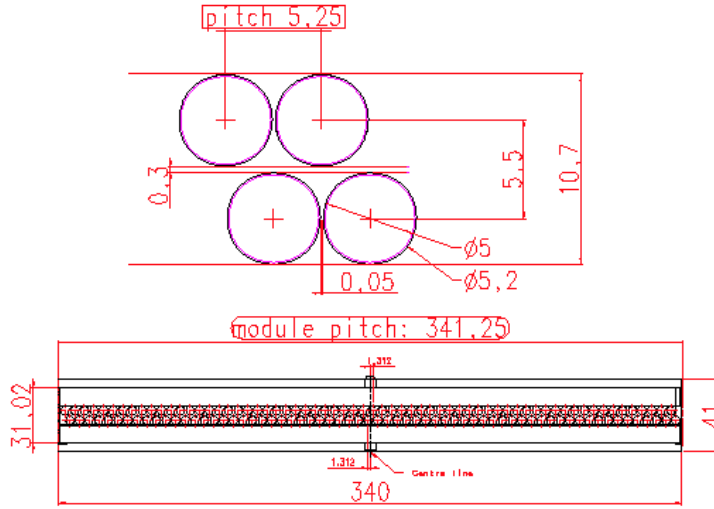


Figura 1.12: Struttura delle camere esterne delle stazioni di tracciamento T realizzate utilizzando camere a deriva. Ogni stazione è realizzata sovrapponendo quattro piani di tubi.

Il tracciamento nella regione esterna

Il sistema di tracciamento esterno (OT) [10] copre la rimanente area delle stazioni traccianti al di fuori dell'accettanza del sistema di tracciamento interno. Il sistema di tracciamento esterno è realizzato mediante tubi a deriva (*strawtubes*) del diametro di 5 mm, come mostrato in figura (1.12).

Una stazione impiega quattro camere sovrapposte di tubi, le quali forniscono misure ridondanti (elevata efficienza) delle coordinate nel piano trasversale. I piani di tubi a deriva hanno i fili orientati alternativamente nella direzione verticale e stereo $\pm 5^\circ$, come mostrato in figura (1.13).

Il tempo di risposta delle camere a deriva dipende oltre che dalla dimensione delle celle (fissata dalla richiesta di ridotta occupanza), anche dalla miscela di gas utilizzata. La miscela $\text{Ar}/\text{CF}_4/\text{CO}_2$ consente di limitare il tempo di risposta a circa 40 ns. Considerando l'integrale di campo e la precisione del sistema di tracciamento, la risoluzione spaziale sulla singola *hit* è dell'ordine di 200 μm . L'errore relativo che ci si aspetta sulla misura dell'impulso è $(0.2 \div 0.4)\%$.

I tipi di tracce

All'interno del rivelatore le particelle cariche possono avere traiettorie molto diverse in base al loro punto d'origine al loro momento, seguendo la figura (1.14) possono essere classificate nel modo seguente:

- **Tracce VELO:** sono le tracce che attraversano solo il velo, in genere sono quelle prodotte a grandi angoli polari o all'indietro e permettono un'accurata determinazione del vertice primario.
- **Tracce UPSTREAM:** sono le tracce che attraversano solamente il velo e le stazioni TT. Sono caratterizzate prevalentemente da un basso impul-

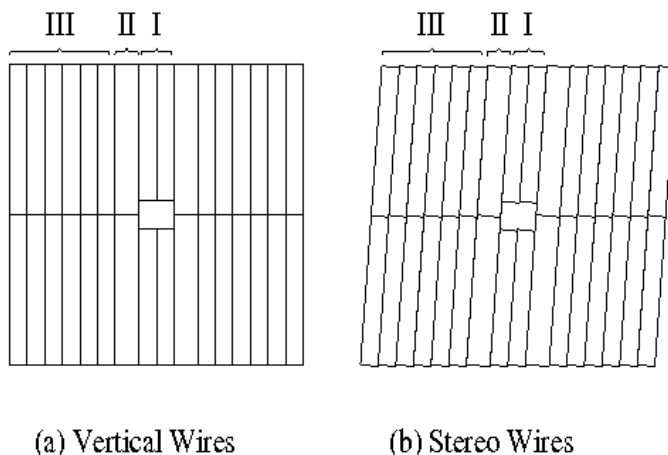


Figura 1.13: Orientazione delle camere a deriva nella regione esterna delle stazioni di tracciamento.

so e sono deflesse dal campo magnetico prima di raggiungere il magnete. A causa della scarsa risoluzione in impulso sono scarsamente usate nell'analisi fisica.

- **Tracce DOWNSTREAM:** Attraversano soltanto le stazioni T e TT senza *hit* nel velo. Sono utilizzate principalmente per la ricostruzione delle particelle decadute all'esterno dell'accettanza del velo (ad es. k_s^0 , Λ).
- **Tracce T:** Sono le tracce misurate solamente dalle stazioni T, tipicamente prodotte dalle interazioni secondarie e sono utilizzate nella ricostruzione nel RICH2.
- **Tracce LONG:** Attraversano tutto il rivelatore dal velo alle stazioni T. Per queste è possibile stabilire con molta precisione i parametri di traccia. Sono le tracce più importanti per la selezione dei decadimenti dei mesoni B.

1.2.6 Il sistema dei RICH

La discriminazione π/K in LHCb è realizzata per mezzo di due rivelatori RICH (Ring Imaging Cherenkov), progettati per operare efficientemente la discriminazione fino a impulsi dell'ordine di 150 GeV/c.

L'identificazione delle particelle mediante i RICH è basata sulla rivelazione dei fotoni di luce Cherenkov, prodotti dal passaggio delle particelle cariche all'interno del mezzo radiatore, quando la velocità delle particelle è maggiore della velocità della luce nel mezzo. L'angolo di emissione ϑ di emissione dei fotoni Cherenkov, misurato rispetto alla direzione di propagazione della particella, dipende dalla velocità v della particella, secondo la relazione $\cos \vartheta = 1 / \left(\frac{v}{c} n \right)$, dove n è l'indice di rifrazione del mezzo e c la velocità della luce nel vuoto.

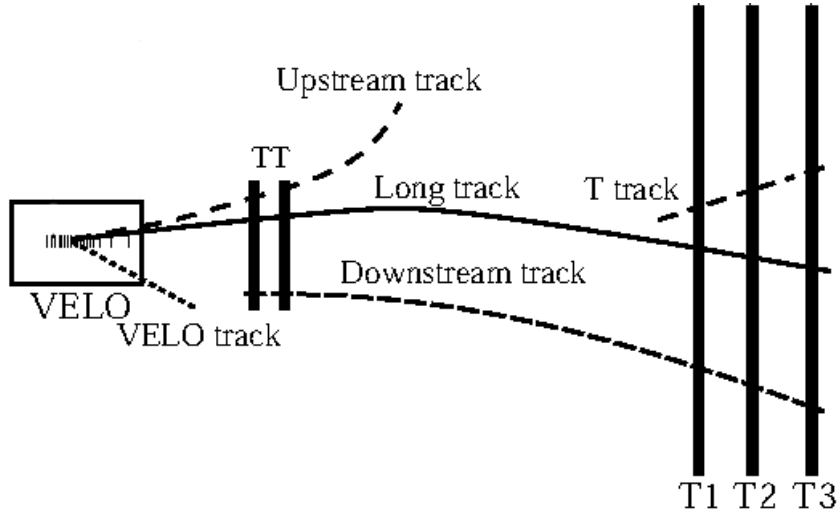


Figura 1.14: Classificazioni delle tracce ricostruite sul piano di deflessione (x, z).

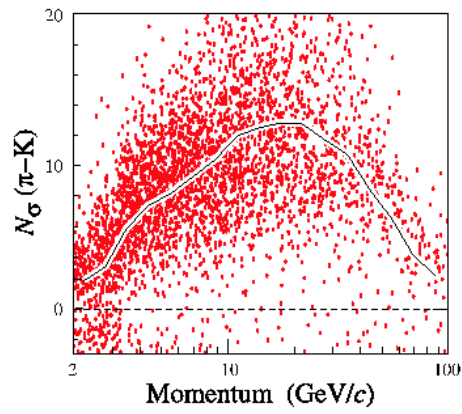


Figura 1.15: La σ della separazione π/K in funzione del momento. La linea continua rappresenta la separazione media.

La figura (1.15) mostra la separazione fra π/K con impulso inferiore a 60 GeV/c espressa in funzione della significanza della separazione π/K in funzione dell'impulso [11]. La significanza è definita come:

$$N\sigma = \sqrt{2|\Delta \ln \mathcal{L}|} \quad (1.5)$$

dove il termine $\Delta \ln \mathcal{L}$ rappresenta il logaritmo della funzione di massima verosimiglianza della differenza fra candidati π e K .

Particelle con un impulso inferiore a 60 GeV/c saranno identificate dal RICH1, le particelle attraversano prima 5 cm di aerogel di silicio poi uno spessore di 85 cm di gas di C_4F_{10} . Particelle di momento maggiore, fino a circa 150 GeV/c saranno identificate dal RICH2, dove le particelle attraverseranno 1.8 m di CF_4 .

Il RICH1, rappresentato in figura (1.16), è collocato in prossimità della regione di interazione, subito dopo il rivelatore di vertice in grado di coprire tutta l'accettanza angolare del rivelatore, le sue dimensioni sono approssimativamente $2 \times 3 \times 1 \text{ m}^3$. RICH1 è costituito da specchi sferici con raggio di curvatura di 2.4 m circa, per focalizzare i fotoni Cherenkov sui foto-rivelatori collocati sui due piani focali. I coni della luce Cherenkov, per effetto della riflessione sugli specchi sferici, appaiono nel piano focale come anelli circolari. I foto rivelatori sono dispositivi HPD (Hybrid Photon Detector) situati nella regione schermata fuori dall'accettanza del rivelatore, dove il campo è minore.

In figura (1.17) è rappresentato uno schema del RICH2. Ha un'accettanza angolare ridotta che si estende fino a 120 mrad nel piano (x, z) e 100 mrad nel piano (y, z) .

1.2.7 I calorimetri

I calorimetri [12] del rivelatore LHCb, come mostrati in figura (1.18), sono posizionati prima delle camere a muoni. Sono usati per rivelare posizione ed energia degli adroni, fotoni, π^0 ed e^\pm .

I depositi energetici delle particelle che colpiscono sono misurati dai calorimetri come risultato dei processi di ionizzazione indotti dalla cascate di particelle secondarie prodotte da questa particella fino all'assorbimento. Tutta la luce emessa nel materiale scintillante, dalla cascata di particelle, viene raccolta e rappresenta una misura dell'energia della particella.

I *cluster* nei calorimetri sono usati dal *trigger* di livello 0 per identificare gli oggetti a più grande massa trasversale come carattere identificativo dei decadimenti del b . Sono usati anche per identificare elettroni e particelle neutre (γ, π^0).

Il calorimetro elettromagnetico (ECAL)

Il calorimetro elettromagnetico è di tipo Shashlik⁵; in esso si alternano materiale scintillatore, spesso 4 mm, con materiale convertitore (piombo), di 2 mm di spessore. ECAL rivela elettroni e fotoni per mezzo della loro cascata elettromagnetica di coppie e^+e^- e γ .

La segmentazione trasversale di ECAL, rappresentata in figura (1.19) consente di ottenere una risoluzione energetica $\sigma(E)/E = 10\%/\sqrt{E} \oplus 1.5\%$, con

⁵Un Shashlik è un particolare tipo di kebab molto venduto al tempo dell'unione sovietica, dove si alternano pezzi di carne a pezzi di grasso.

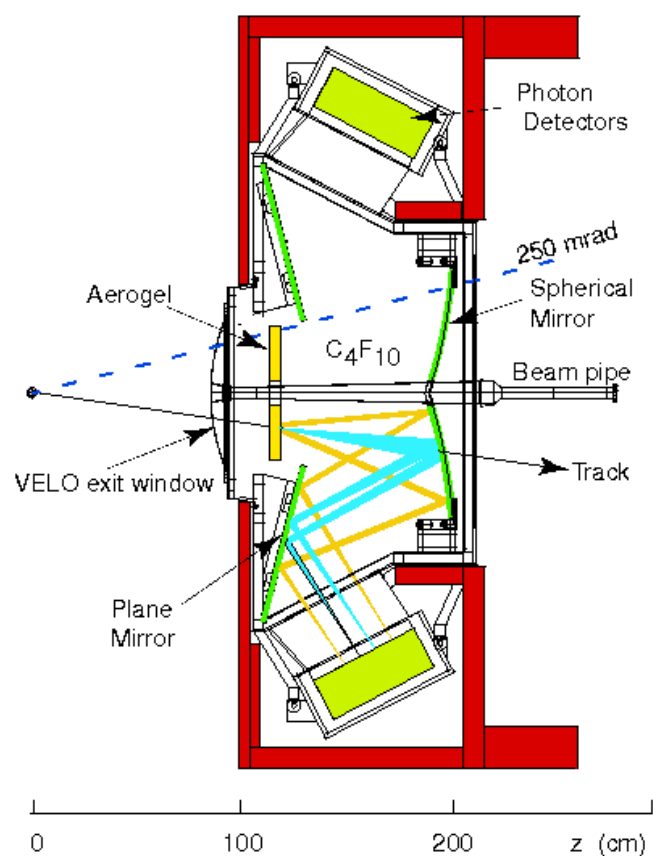


Figura 1.16: Veduta sul piano (x, y) del rivelatore Cherenkov RICH1. Poiché il rivelatore sarà immerso in un campo magnetico, il progetto prevede la collocazione dei foto-rivelatori in una regione schermata. La luce Cherenkov è convogliata ai foto-rivelatori mediante l'utilizzo di specchi piani che effettuano una riflessione secondaria.

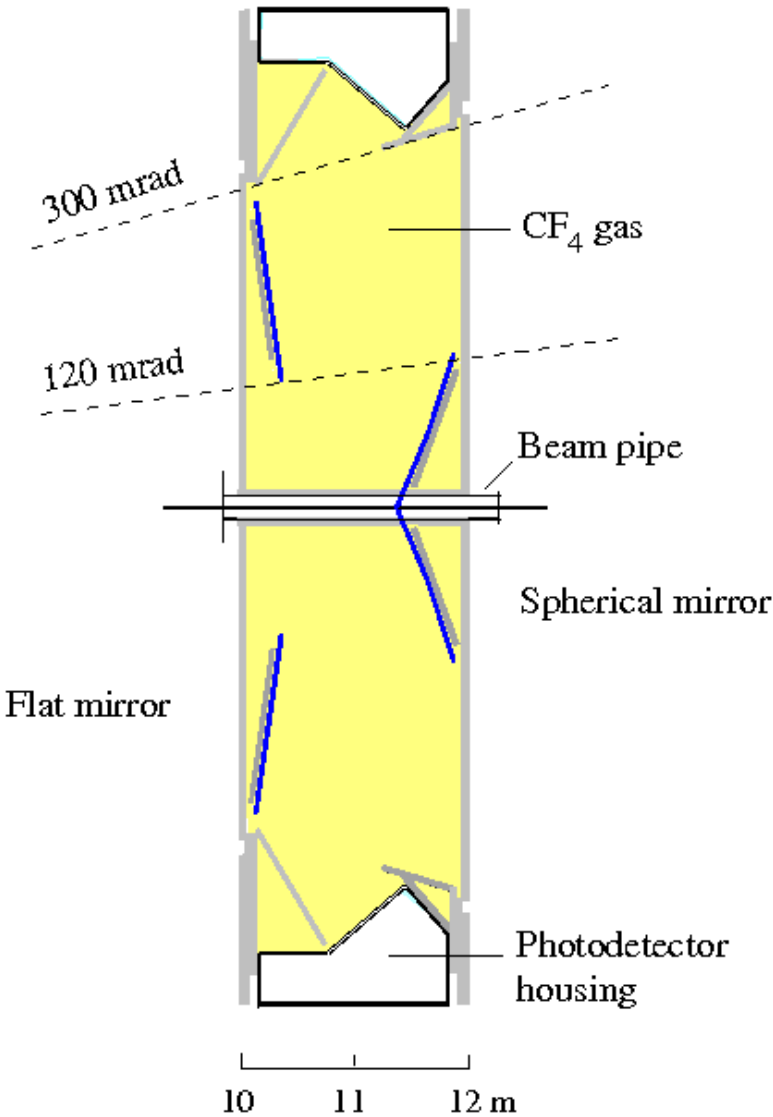


Figura 1.17: Veduta schematica del rivelatore RICH2 nel piano (x, y) . Il RICH2 è usato per la discriminazione π/K ad alto impulso nella regione a piccolo angolo.

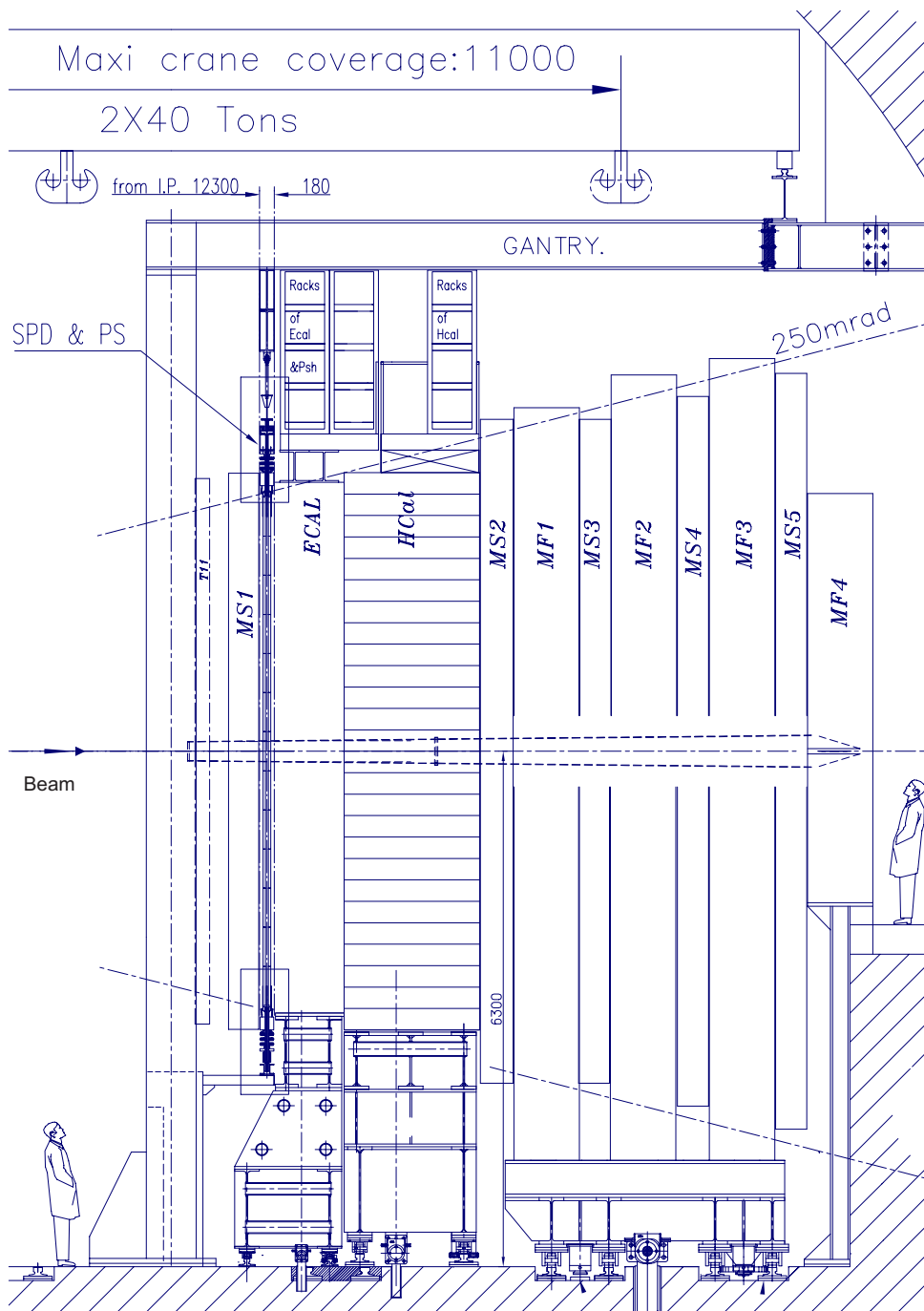


Figura 1.18: Schema dei calorimetri di LHCb, nel disegno sono rappresentate anche le camere a muoni.

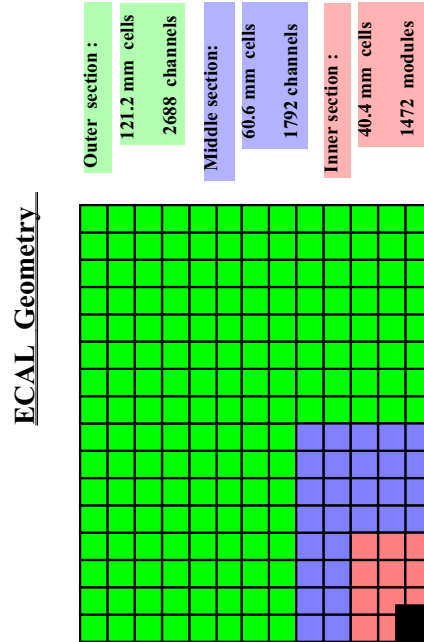


Figura 1.19: Segmentazione del calorimetro ECAL.

E espressa in GeV, dove: il primo termine, stocastico, descrive le fluttuazioni statistiche nel rilascio energetico dovuto alla cascata elettromagnetica, mentre il secondo termine, deterministico, rappresenta gli effetti sistematici indotti dal rivelatore.

Scintillating Pad Detector (SPD)

L'SPD, fotografato frontalmente nella figura (1.20), permette di identificare le particelle cariche attraverso la loro ionizzazione su un piano di scintillatore dello spessore di 15 mm. Questo rivelatore è usato in coincidenza con ECAL per distinguere fra fotoni ed elettroni. La luce prodotta dagli scintillatori è raccolta da fibre ottiche e diretta a fotomoltiplicatori multi-anodo.

Pre-Shower (PS)

Una lastra di piombo dello spessore di 12 mm innesca la cascata elettromagnetica che viene rivelata dal PS. Anch'esso è costituito da piani di scintillatore come l'SPD e ha lo scopo di permettere di distinguere fra elettroni e adroni.

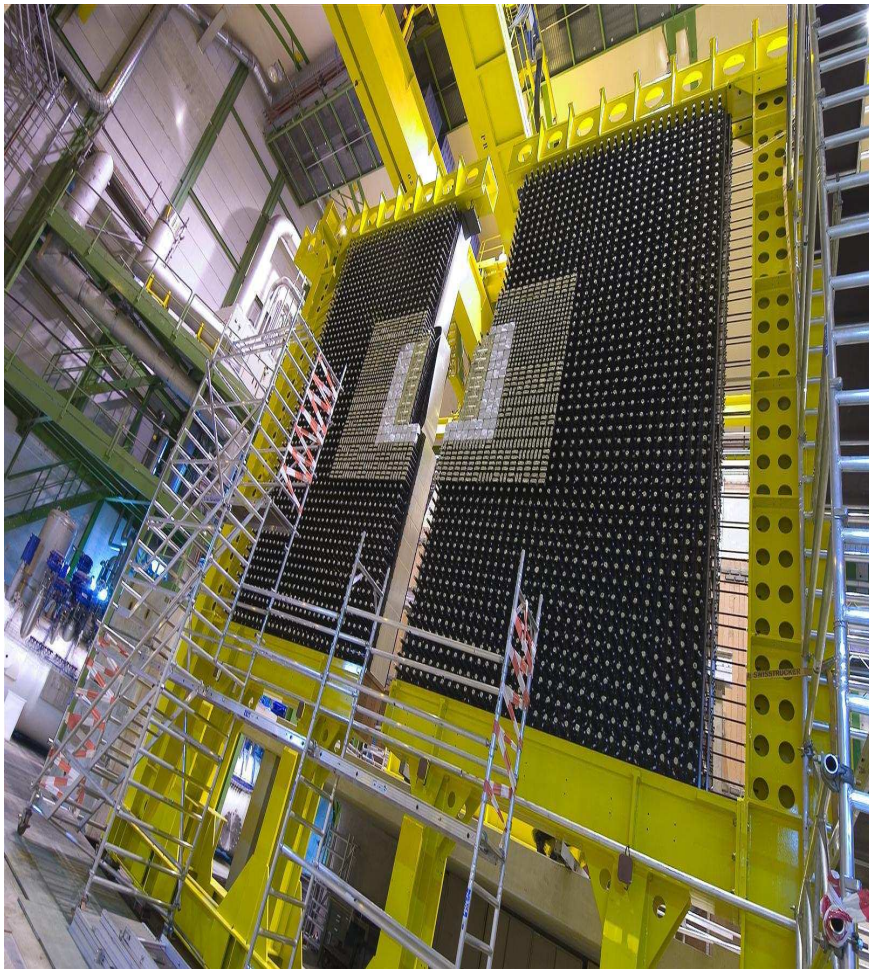


Figura 1.20: Immagine frontale dei calorimetri SPD/PD di LHCb.

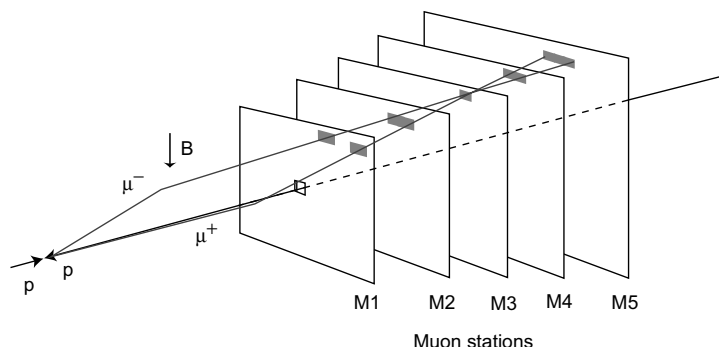


Figura 1.21: Il sistema delle camere a muoni è utilizzato dal *trigger* per l'identificazione delle particelle μ .

Il calorimetro adronico HCAL

Il calorimetro adronico è collocato oltre il calorimetro elettromagnetico ed è usato per identificare gli adroni tramite la loro interazione con il materiale del rivelatore, nella sola fase di *trigger* L0. È composto da piani di ferro, spessi 16 mm, e materiale scintillatore, spesso 4 mm, alternati, disposti parallelamente alla direzione del fascio. La risoluzione in energia attesa è piuttosto scarsa ma adeguata agli scopi del *trigger*: $\sigma(E)/E = 80\%/\sqrt{E} \oplus 10\%$.

1.2.8 Le camere a muoni

I muoni sono identificati tramite un sistema dedicato, disposto oltre i calorimetri (stazioni M2, M3, M4, M5), figura (1.18), con l'eccezione della stazione M1 che è situata prima dell'SPD.

Il rivelatore di muoni nel sistema di *trigger* è usato con un ruolo primario, come mostrato in figura (1.21). Il sistema delle camere a muoni è anche utilizzato, nella ricostruzione offline, per identificare i muoni, partendo dalle tracce identificate dalle camere traccianti ed estrapolandole fino alle stazioni dei muoni per confermare l'ipotesi di identificazione di un muone.

Le camere a muoni sono costituite prevalentemente di camere proporzionali multi-filo (MWPC), di cui un modulo è mostrato in figura (1.22), divise in quattro regioni con diverse granularità. La parte più interna di M1 a causa dell'alto livello di radiazione richiede l'uso di rivelatori di tipo *triple-GEM*⁶ [13].

1.2.9 Il sistema di trigger

Il sistema di *trigger* di LHCb, descritto nella sua prima versione nel *trigger* TDR [14], prevedeva originariamente tre livelli attivi in cascata. Ha la funzione di selezionare gli eventi di decadimento dei mesoni B di interesse, prodotti nell'accettazione del rivelatore. Il suo progetto è stato semplificato nel 2005 dopo

⁶ Gaseous Electron Multiplier.

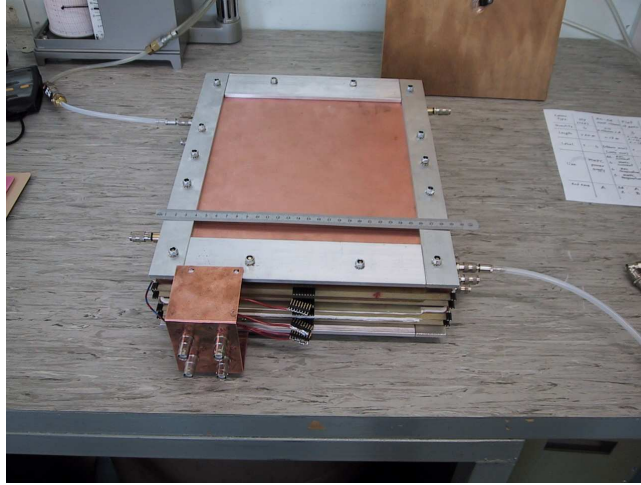


Figura 1.22: Foto di un modulo di camera proporzionale a multi-fili usata per la rivelazione dei muoni.

il “Real Time Trigger Challenge” [15] riducendolo a soli due stadi: il primo livello è realizzato mediante schede elettroniche, il secondo è un *trigger software*, realizzato tramite una *farm* di calcolatori.

La frequenza delle interazioni visibili⁷ alla luminosità di LHCb, è attesa essere di circa 10 MHz, mentre la frequenza di eventi $b\bar{b}$ è 100 kHz, dei quali soltanto il 15% sarà entro l'accettanza di LHCb; i rapporti di diramazione dei canali di interesse per lo studio della violazione di CP sono tipicamente minori di 10^{-3} .

È evidente l'esigenza di disporre di un sistema di *trigger* che permetta di riconoscere gli eventi che potranno essere analizzati offline, rigettando quelli di fondo.

Il trigger L0

Il primo stadio del sistema di acquisizione è realizzato mediante schede elettroniche costruite appositamente e ha la funzione di ridurre la frequenza di acquisizione degli eventi da 40 MHz (10 MHz di interazioni visibili) a 1 MHz, corrispondente alla massima frequenza sostenibile dal livello di *trigger* successivo. I prodotti di decadimento dei mesoni B hanno la caratteristica di avere un alto impulso trasversale rispetto agli eventi di fondo. Il *trigger* L0 opera la selezione degli eventi in base all'energia e all'impulso trasversale:

- adroni, elettroni, fotoni e π^0 con la più alta energia trasversale rivelata dai calorimetri;
- muoni con impulso trasversale più alto rivelati dalle camere per i muoni.

Esso può far uso anche della misura della molteplicità delle tracce cariche realizzata contando le celle sensibili colpite del rivelatore SPD, e pure, come detto

⁷Un'interazione si dice visibile se produce almeno due tracce cariche con un numero sufficiente di hit nel VELO e nelle camere traccianti ed è quindi rilevabile dall'apparato sperimentale.

in precedenza, della misura del numero delle interazioni primarie per escludere eventi eccessivamente complessi, la cui ricostruzione sarebbe troppo dispendiosa in termini di tempo di CPU per il livello di *trigger* successivo. Il tempo di latenza di L0 (sistema completamente sincrono) è fissato in circa 4 μ s.

Il trigger HLT

Il *trigger* di livello superiore (HLT), successivo a L0, è un *trigger software*, realizzato mediante algoritmi di selezione eseguiti su una rete di computer commerciali (*Event Filter Farm*) composta da circa 2000 CPU. Il *trigger* HLT ha la funzione di ridurre la frequenza degli eventi da 1 MHz a 2 kHz, selezionando eventi con beauty e anche segnali di charm.

Benché la frequenza di 1 MHz sia molto elevata vi sono diversi vantaggi nell'utilizzo di un *trigger software*, rispetto a un *trigger hardware* (ovvero un sistema elettronico dedicato per la selezione degli eventi) che possono essere riassunti essenzialmente nei punti seguenti:

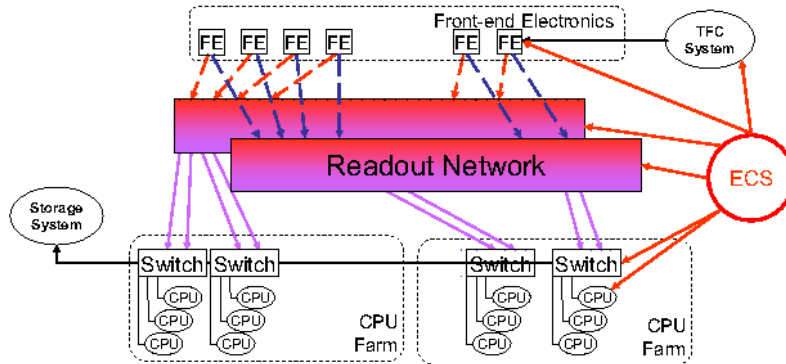
- Flessibilità: i criteri di selezione degli eventi possono essere modificati con un cambiamento dei programmi in esecuzione sui PC della *farm*; nel caso del *trigger hardware* occorre quando possibile riprogettare i circuiti.
- Scalabilità: il rateo degli eventi analizzati può essere aumentato incrementando il numero dei PC (ed eventualmente ridimensionando lo *switch Ethernet*).
- Costo: I personal computer, avendo diffusione di massa, hanno un costo decisamente inferiore a quello di circuiti elettronici progettati ad hoc che eseguano le stesse funzioni. I computer utilizzati nella *farm* sono essenzialmente PC commerciali contenuti in scatole (box 1U) predisposte e dimensionate per il montaggio su rack e dotati di un sistema di alimentazione e di raffreddamento particolarmente affidabili (i *rack* sono raffreddati mediante un circuito ad acqua).

L'utilizzo del trigger software è una tecnica sperimentale consolidata per frequenza significativamente inferiori a quelle in questione, **con questi livelli di frequenza d'ingresso, risulta essere una tecnica innovativa.**

I sotto-rivelatori di LHCb inviano le loro informazioni (frammenti d'evento) al *buffer* del *trigger* HLT realizzato tramite un sistema di schede elettroniche chiamate TELL1⁸. Queste raccolgono le frazioni di evento provenienti dai sotto-rivelatori; tali frazioni sono raggruppate in multi-eventi o MEP (*Multi Event Package*) fino ad un numero di 20 frammenti consecutivi. I MEP sono usati per ottimizzare l'impiego di datagrammi IP ovvero ridurre il peso relativo dello *header* IP rispetto alla parte dei dati effettivamente trasmessa.

Quando sono stati raccolti gli eventi previsti, i MEP sono trasmessi in maniera sincrona attraverso il *Readout Network* utilizzando il protocollo IP da tutte le schede TELL1 a uno dei nodi di calcolo della *farm* che deve avere risorse di CPU disponibili per effettuare l'elaborazione di selezione mediante gli algoritmi di *trigger*.

⁸Attualmente ci sono 276 schede TELL1, ciascuna dotata di 36 porte ottiche d'ingresso e quattro porte Gigabit Ethernet d'uscita

Figura 1.23: Schema del sistema di *trigger software* HLT.

Il *trigger software* non ha latenza fissata. Fondamentale è che il tempo medio impiegato per l'elaborazione dell'algoritmo di selezione rientri nel periodo che separa l'ingresso di due eventi successivi nel sistema di *trigger*. Il tempo di calcolo medio per l'esecuzione dell'algoritmo HLT ritenuto adeguato è dell'ordine di 2 ms. Considerando che mediamente un evento ha dimensioni di 50 kB, il flusso di dati dalle schede TELL1 ai nodi della *farm* è di circa 50 GB/s[15].

L'architettura del sistema è mostrata in figura (1.23). I nodi di calcolo sono raccolti in 50 rack che ospitano più di 40 PC ciascuno: un modulo è mostrato nella foto della figura (1.24). Il numero complessivo dei PC necessari per realizzare la *farm* è dato dalla richiesta di elaborare i dati in ingresso con una frequenza ν di 1 MHz con un tempo T medio di 2 ms, secondo la formula:

$$\langle T \rangle = \text{nr. CPU} \times \frac{1}{\nu} \quad (1.6)$$

Lo switch usato per la realizzazione della rete è un Single core router (Force10 E1200, 1260 GbE ports) che si può vedere nella foto (1.25).

L'algoritmo di selezione del secondo livello di *trigger* è attuato in sintesi effettuando i seguenti passaggi:

- Verifica rapida della decisione prodotta dal livello di *trigger* L0 precedente.
- Ricostruzione dei vertici secondari utilizzando le tracce bidimensionali definite nel piano (r, z) , usando i sensori R del VELO. Calcolo del parametro d'impatto di ciascuna traccia bidimensionale rispetto al vertice primario. Le tracce con elevato parametro d'impatto sono successivamente ricostruite in tre dimensioni, utilizzando anche le coordinate angolari dei sensori φ . L'impulso delle tracce è misurato estrapolando i segmenti di traccia ricostruiti nel rivelatore di vertice fino alla camera TT. In questa zona è presente un campo magnetico residuo avente un integrale di campo noto di 0.15 Tm. La precisione sul parametro d'impatto è dell'ordine di 40 μm e la precisione della misura dell'impulso, in questa fase è di circa il 20%.
- Ricostruzione completa dell'evento che è attuata a tutti gli eventi che hanno superato i due passaggi precedenti, dove molto probabilmente sono stati prodotti mesoni B, per selezionare modalità di decadimento specifiche.

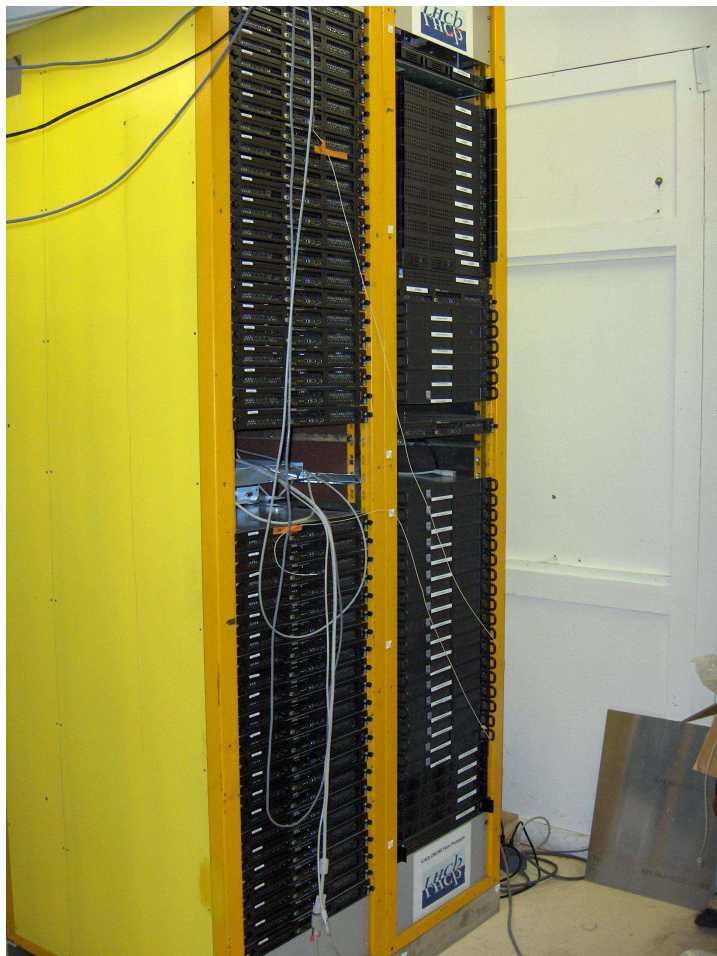


Figura 1.24: Un rack di PC utilizzato per il *trigger* online che compongono l'Event Filter Farm.



Figura 1.25: Switch con 1260 porte Gigabit Ethernet, impiegato per la realizzazione del *trigger* HLT.

Il funzionamento di HLT si basa su ipotesi o percorsi di decadimento del mesone B prodotto, come schematizzato in figura (1.26).

HLT presenta un percorso per i muoni, uno per muoni e adroni, uno per adroni e uno per particelle elettromagnetiche. Ciascuno di questi viene eseguito in base alle decisioni prese da L0 nella selezione delle particelle candidate. Tale sequenza può includere sia selezioni esclusive che inclusive e una classificazione per eventi fisici di interesse.

Per prima cosa, nei percorsi, viene eseguita una ricostruzione parziale dell'evento. Durante questa ricostruzione parziale, poche tracce sono selezionate in base al loro momento trasversale e parametro d'impatto (rispetto al vertice primario). La ricostruzione completa viene eseguita solo allo stadio finale dell'algoritmo completo, dopo una significativa diminuzione del rateo degli eventi. La maggior parte del tempo viene impiegata per la ricostruzione di vertici e tracce. La precisione raggiunta sui vertici e sulle tracce è confrontabile con quella che si potrà ottenere successivamente dall'analisi *off-line*. Ad esempio, per tracce lunghe che attraversano tutto il rivelatore si ottiene una precisione sul momento che vale $\sigma_p/p \approx 1\%$ mentre nella ricostruzione *off-line* la precisione è circa il 0.4%.

- Il **percorso adronico**, schematizzato in dettaglio nella figura (1.27) viene eseguito se il *trigger* L0 viene superato da un candidato adronico. In un primo stadio di "*pre-trigger*", le tracce allineate nel rivelatore di vertice sono ricostruite e selezionate se il loro parametro d'impatto supera la condizione $IP > 150 \mu\text{m}$. Il vertice primario è ricostruito con una precisione di $\sigma_z \approx 60 \mu\text{m}$ lungo la direzione del fascio e di circa $\sigma_{x,y} \approx 20 \mu\text{m}$

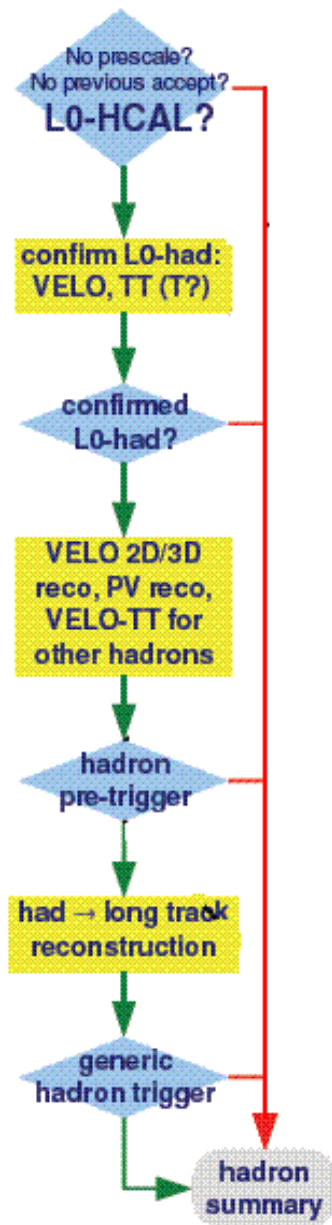


Figura 1.27: Dettaglio dei vari passaggi di analisi effettuate da HLT durante l'esecuzione dell'ipotesi di decadimento adronica.

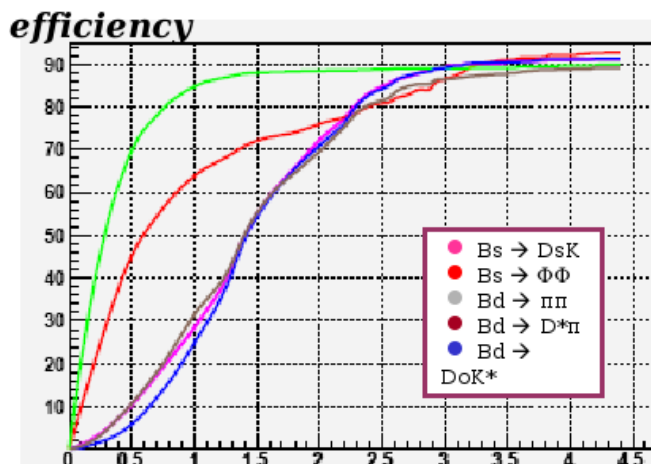


Figura 1.28: Prestazioni della selezione adronica di HLT tramite l'uso delle ipotesi o percorsi di decadimento.

2.5 GeV (non sono applicati tagli sui parametri d'impatto) gli eventi che passano HLT vengono salvati.

L'evento supera HLT anche se la massa invariante è superiore a 0.5 GeV e il parametro d'impatto soddisfa la disuguaglianza $|IP| > 100 \mu\text{m}$. Eventi con un singolo muone passano il percorso muonico se $p_T > 3.0 \text{ GeV}$ e la significatività del parametro d'impatto $IP/\sigma_{IP} > 3\sigma$. Complessivamente gli eventi del percorso adronico passano HLT con un rateo di circa 1.5 kHz.

- **Il percorso adronico e muonico** viene eseguito quando il livello 0 è stato passato da un candidato muone (o di muone) ma il muone di HLT non supera il percorso muonico di HLT. In questo caso, il muone non ha sufficiente p_T per superare il percorso muonico, la strategia è di selezionare eventi con una traccia di muone e una di adrone, con alto parametro d'impatto e impulso trasversale relativamente alto. Si valuta di poter ottenere una purezza del b attorno al 50-60% per un rateo HLT attorno a $2 \div 5 \text{ kHz}$
- **Il percorso elettromagnetico** L0 fornisce circa 200 kHz di candidati selezionati dal calorimetro elettromagnetico. Questi candidati elettroni e fotoni sono per prima cosa confermati nello stadio di *pre-trigger* del percorso elettromagnetico. Per gli elettroni viene anche richiesto che la traccia abbia parametro d'impatto compatibile con quella del candidato L0. Per i candidati fotoni è previsto di richiedere, in aggiunta, l'esistenza nell'evento di un adrone con grande impulso trasversale e grande parametro d'impatto rispetto al vertice primario. Questa strategia deve ancora essere completata e ha lo scopo di migliorare l'efficienza di alcuni canali come $B^0 \rightarrow K^* \gamma$.

Una ricostruzione di tutte le tracce viene eseguita per ogni selezione inclusiva a pochi kHz. Le selezioni inclusive più comuni sono per D_S , D^* , Φ e muoni. Ad esempio la selezione inclusiva di D^* fornisce una statistica molto alta del

campione di eventi $D^{*+} \rightarrow D^0(K^-\pi^+)\pi^+$. Si prevede che questa selezione adronica sarà eseguita con un rateo di poche centinaia di Hz.

- **La selezione inclusiva** dei muoni sfrutta la presenza o di un singolo muone o di una coppia di muoni. Nel caso della coppia proveniente da $B \rightarrow J/\psi(\mu^+\mu^-)K_S$ il decadimento è ricostruito con una risoluzione in massa invariante di 17 MeV[16]. Evitando un taglio sul parametro d'impatto si permette la selezione di di-muoni senza *bias* della vita media del mesone B che le ha prodotte. La selezione inclusiva di un singolo muone fornisce una purezza molto alta, vicina al 70% del campione $B \rightarrow \mu X$ che può servire per studiare il rendimento nel riconoscimento (*tagging*) delle particelle prodotte.
- Alla fine della sequenza HLT rimangono attive ancora poche **selezioni dei decadimenti esclusivi** dei mesoni B che vengono realizzati sfruttando le particelle selezionate (D_S , D^* , K^*) nei percorsi precedenti. Vengono in genere applicati tagli in massa molto larghi ≈ 500 MeV. Queste selezioni esclusive ottengono un'efficienza approssimativamente del 90% per un rateo d'uscita dell'ordine di 200 Hz.

Capitolo 2

Il sistema di monitor e di controllo

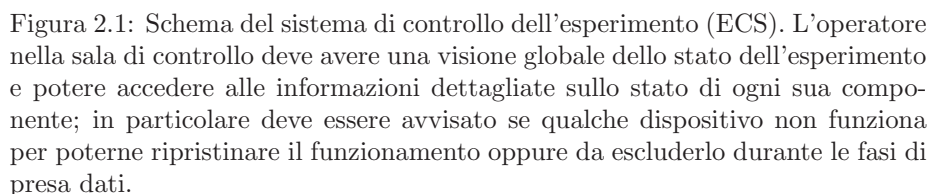
2.1 Il sistema di controllo dell'esperimento LHCb (ECS)

Il sistema di controllo dell'esperimento LHCb (ECS, Experimental Control System) è stato progettato per **configurare**, **controllare** e **monitorare** il funzionamento di tutte le componenti *on-line* dell'apparato sperimentale, come mostrato nella figura 2.1, in particolare:

- il sistema di acquisizione dati e di *trigger*: la temporizzazione, l'elettronica di *front-end*, l'elettronica del *trigger* di livello zero, la rete dati di *readout*, la *farm* di computer dedicata alla selezione degli eventi di interesse (EFF, Event Filter Farm), ecc.;
- il rivelatore (DCS, Detector Control System): il flusso e la pressione dei gas, le alimentazioni ad alta e bassa tensione, le temperature, ecc.;
- l'infrastruttura sperimentale: il magnete, il raffreddamento e la ventilazione, la distribuzione della potenza elettrica, il sistema di sicurezza del rivelatore, ecc.;
- l'interazione con sistemi di controllo esterni: l'acceleratore LHC, il sistema di sicurezza e i servizi tecnici del CERN.

Il sistema di controllo di LHCb è stato sviluppato nell'ambito del progetto JCOP (Joint COntrols Project [18]) del CERN, costituito per uniformare e integrare il più possibile i sistemi di controllo dei quattro esperimenti al LHC.

L'ECS di LHCb è realizzato tramite un sistema SCADA (Supervisory Control and Data Acquisition) di origine commerciale, denominato PVSS (Prozeß-visualisierungs und Steuerungssystem) [17], che permette di realizzare interfacce grafiche contenenti tabelle, grafici, pulsanti, led, menu di scelta di vario tipo, mediante le quali è possibile interagire con i dispositivi dei sotto-rivelatori di LHCb per modificarne le configurazioni, attivare o disattivare allarmi, escludere parti del sistema che non funzionano correttamente durante l'acquisizione dati.



Nell'ECS i data-point rappresentano i sensori delle grandezze sotto controllo (p. es. tensioni di alimentazione, traffico di rete, temperature) e sono rappresentati sui pannelli — pure realizzati con PVSS — che ne consentono una visualizzazione tabellare oppure grafica (per evidenziarne l'andamento temporale). Due esempi di pannelli sono nelle figure 2.3 e 2.4.

Il sistema gerarchico è stato implementato mediante una macchina a stati finiti (FSM, Finite State Machine [20]) descritta mediante il linguaggio SML (State Machine Language) e costituita (si veda la figura 2.5) da due tipi di *unità* logiche:

- *device unit*, unità direttamente a contatto con i dispositivi di rivelazione, il cui stato dipende direttamente dai parametri fisici misurati;
- *control unit*, unità di grado gerarchicamente più elevato, il cui stato è funzione dello stato delle *control unit* o delle *device unit* direttamente

2.1. IL SISTEMA DI CONTROLLO DELL'ESPERIMENTO LHCb (ECS)37

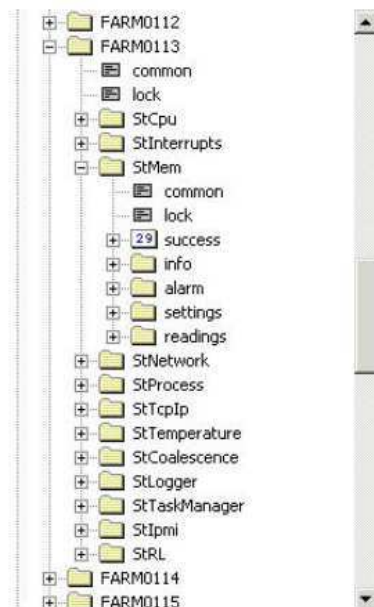


Figura 2.2: Struttura dei data-point per un nodo di calcolo della *farm*.

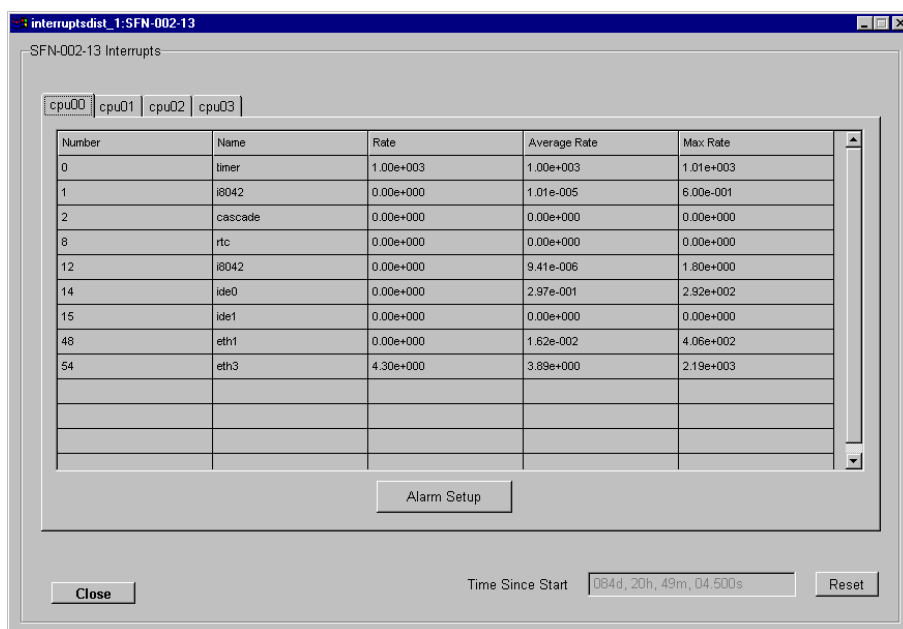


Figura 2.3: Pannello PVSS che visualizza i ratei di *interrupt hardware* sui nodi di calcolo della *farm*.

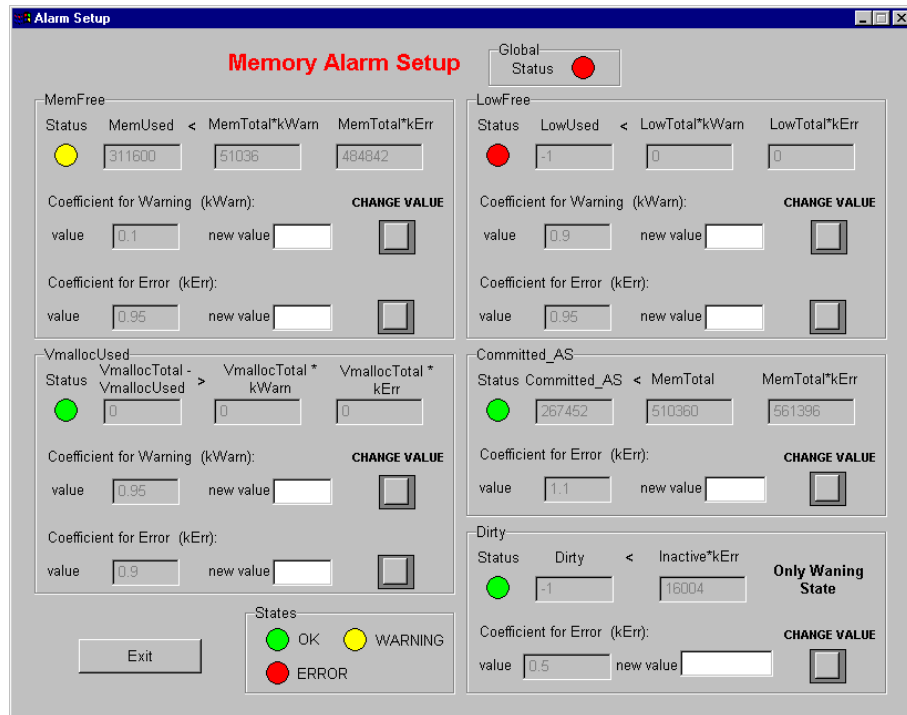


Figura 2.4: Interfaccia grafica realizzata mediante PVSS per configurare le soglie di allarme relative allo stato della memoria di un nodo della *farm*.

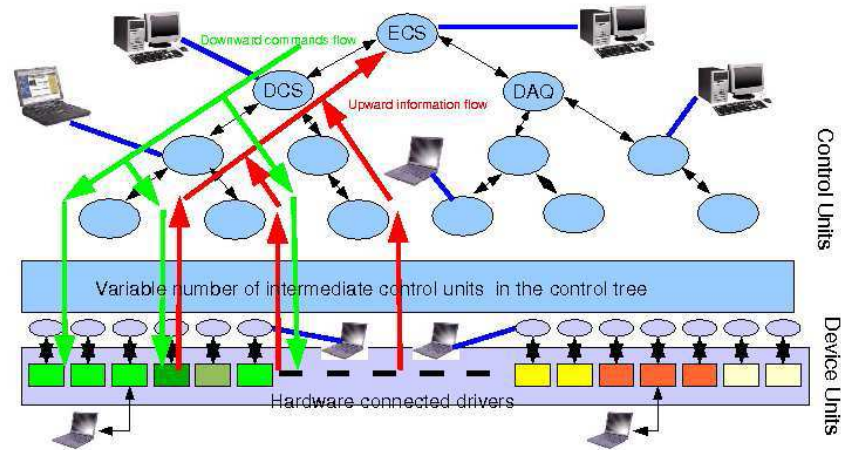


Figura 2.5: Schema generale dell'organizzazione dell'ECS: ogni elemento del sistema propaga le informazioni sul proprio stato tramite una macchina a stati finiti. Lo stato complessivo del sistema dipende dallo stato di ogni suo elemento costitutivo.

Colore	Stato
ROSSO	Errore
GIALLO	Non pronto
BLU	Pronto
VERDE	In esecuzione

Tabella 2.1: Tabella delle associazioni colore-stato convenzionalmente adottate nell'ECS.

subordinate.

Allo stato di ogni unità di controllo è associato anche un colore, seguendo la convenzione riportata nella tabella 2.1.

Lo stato dell'unità di massimo livello gerarchico dipende perciò logicamente — in ultima analisi — dallo stato delle *device unit*.

Nella struttura gerarchica ad albero, un sotto-albero in stato di errore può essere separato dalla struttura e controllato separatamente (*partitioning*), per isolare la *device unit* o il gruppo di *device unit* che determina lo stato di errore e consentendo nel frattempo di riportare il resto della struttura nello stato di operatività.

Nella figura 2.6 alcuni nodi sono in stato di errore (marcati in colore rosso) e possono essere esclusi automaticamente o manualmente (i nodi esclusi sono marcati in grigio). Selezionando uno di questi nodi appare il pannello con lo stato delle grandezze controllate (memoria, cpu, rete, temperature, ecc.). È possibile selezionare ciascuna di queste grandezze per visualizzare tutte le informazioni relative nel pannello PVSS nello strato gerarchico più basso.

2.2 Il sistema di monitor e controllo della farm on-line (FMC)

Il sistema di *trigger* di alto livello dell'esperimento LHCb (HLT, High Level Trigger) è realizzato da una *farm* di calcolatori — denominata Event Filter Farm (EFF) — composta da circa 2000 PC interconnessi mediante una rete commutata Gigabit Ethernet, ed è stato progettato per essere scalabile e facilmente potenziabile. Secondo le caratteristiche del progetto, esso deve gestire un flusso di dati in ingresso di circa **50 GiB/s**¹ [15] riducendo il rateo degli eventi da 1 MHz a 2 kHz.

Poiché l'EFF deve elaborare *on-line* un enorme flusso di dati, i processi di *trigger* in esecuzione sui nodi di calcolo della *farm* dovranno avere a loro completa disposizione le risorse di calcolo (CPU) disponibili e faranno un uso intenso dei *driver* delle interfacce di rete del *kernel* e dello *stack* TCP/IP.

La EFF richiederà perciò un **sistema di monitor** in grado di sovrintendere dettagliatamente all'attività dei nodi di calcolo, controllando nel particolare il valore dei parametri di funzionamento della CPU, dell'interfaccia di rete Gigabit Ethernet e dell'attività dello *stack* TCP/IP [21] e un **sistema di controllo** in

¹Nello svolgimento della tesi, seguendo le norme IEC 60027-2, seconda edizione, 2000-11, per evitare ambiguità si utilizzano i prefissi Ki, Mi e Gi per indicare 2^{10} , 2^{20} e 2^{30} , preservando per i prefissi k, M e G il significato originale nel Sistema Internazionale di 10^3 , 10^6 e 10^9 [19].

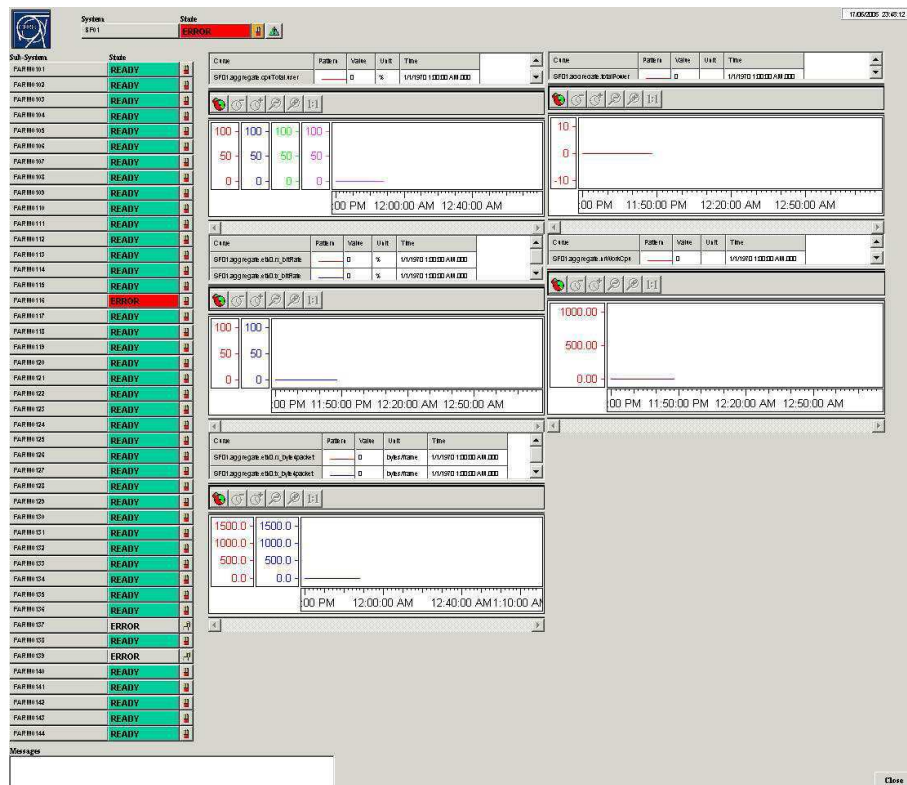


Figura 2.6: Pannello della macchina a stati finiti per la *farm* dei calcolatori del *trigger on-line*. Alcuni nodi di calcolo sono in stato di errore e possono essere esclusi sia automaticamente che manualmente dall'analisi *on-line* (in grigio i nodi esclusi).

2.2. IL SISTEMA DI MONITOR E CONTROLLO DELLA FARM ON-LINE (FMC)41

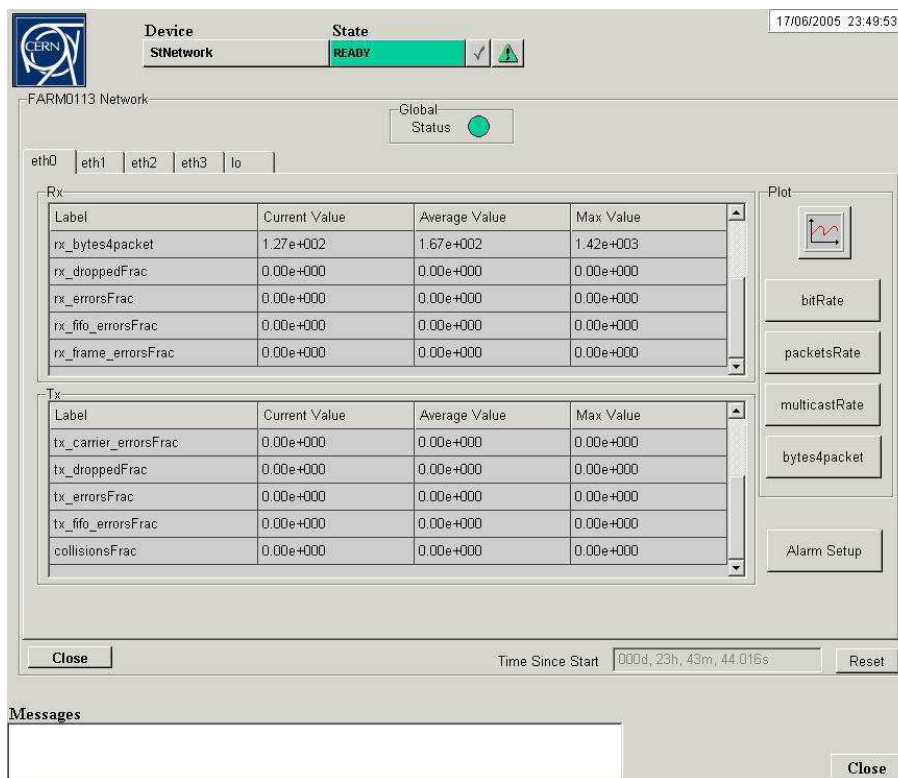


Figura 2.7: Esempio di un pannello relativo ai sensori della rete di un nodo di calcolo della *farm*.

grado di sfruttare senza inconvenienti² lo *scheduling real-time* dei processi, previsto nel sistema operativo Linux nella modalità *FIFO* (First In First Out) o *Round Robin*, in luogo dello *scheduling* standard di Linux, il classico *scheduling* UNIX, di tipo *time-sharing*³.

2.3 Descrizione del sistema FMC

Il sistema di Monitor e di Controllo della Event Filter Farm (FMC, Farm Monitoring and Control System) è stato integralmente **progettato e realizzato** dal gruppo LHCb del **Dipartimento di Fisica dell'Alma Mater Studiorum – Università di Bologna** e della **Sezione di Bologna dell'INFN**.

Pensato inizialmente per l'uso esclusivo della collaborazione LHCb, il *software* è stato successivamente integrato nel progetto JCOP (si veda la pagina web “Farm Monitoring and Control” presso la IT Division del CERN [22]): alcune componenti del sistema FMC sono già state adottate anche dagli esperimenti Atlas e CMS, mentre altre componenti sono attualmente sottoposte a valutazione da parte degli esperimenti Atlas, CMS e Alice.

Il sistema FMC è composto dalle seguenti componenti *software*, che saranno descritte in maggior dettaglio nelle sezioni successive:

- L'*Event Logger* raccoglie i messaggi diagnostici spediti dalle applicazioni in esecuzione sui nodi della *farm*.
- Il *Power Manager* può spegnere e accendere i nodi della *farm*; inoltre può registrarne i parametri fisici di funzionamento (temperature, velocità delle ventole, tensioni di alimentazione, consumo elettrico, ecc.).
- Il *Task Manager* è in grado di avviare e terminare i processi sui nodi della *farm* da una console centrale, e dispone di un meccanismo di notifica real-time della terminazione inaspettata dei processi.
- Il *Process Controller* mantiene in esecuzione una lista di applicazioni sui nodi della *farm* e, se richiesto — interagendo con il Task Manager — permette il riavvio automatico (re-spawning) dei processi inaspettatamente terminati.
- Il *Monitor System* controlla i parametri di funzionamento dei nodi della *farm*.

A eccezione del *Process Controller* — che dipende dal *Task Manager* — tutti gli altri strumenti sopra elencati sono indipendenti l'un l'altro e possono essere utilizzati singolarmente. Tutte le componenti dell'FMC possono inviare i loro messaggi diagnostici all'*Event Logger* (ma possono essere configurate per inviare i messaggi allo *standard error stream* o al servizio di *system log* del sistema operativo). Il *Task Manager* è in grado di reindirizzare all'*Event Logger* lo *standard output stream* e lo *standard error stream* dei processi avviati.

Essendo la **tolleranza agli errori** (*fault-tolerance*) un importante requisito del sistema, il *software* è stato progettato seguendo i seguenti criteri:

²Un processo *real-time ill-behaved* può portare al blocco totale del sistema operativo in assenza di un processo di controllo, in esecuzione a priorità più alta, in grado di terminarlo.

³Si veda il punto 7 a pagina 55 per una descrizione degli *scheduler* di Linux.

- cercare di ottenere il codice più semplice possibile;
- sfruttare — fintanto che è possibile — i servizi già esistenti nel *kernel* del sistema operativo Linux, in quanto essi sono sicuramente affidabili e ottimizzati;
- gestire e registrare il verificarsi di qualsiasi condizione eccezionale ritenuta possibile (anche se improbabile);
- garantire la sopravvivenza del sistema FMC al riavvio di qualunque suo componente⁴.

Il sistema FMC è composto da un insieme di *server* “leggeri” (*back-end*) in esecuzione sui nodi della *farm* che comunicano per mezzo dello strato *software* di comunicazione DIM (Distributed Information Management System [23]), con un numero arbitrario di *client* (*front-end*) in esecuzione su PC diversi. I *client* sono disponibili sia come processi Linux interattivi con interfaccia testuale a riga di comando, sia come processi PVSS inter-piattaforma (Linux/MS Windows), interattivi, con interfaccia di tipo grafico (GUI).

In particolare, i *client* grafici PVSS interfacciano il sistema FMC con la macchina a stati finiti (FSM) dell’esperimento, inducendo la transizione di una sua *unità* a uno stato di allarme ogniqualvolta i parametri monitorati indicano condizioni di lavoro anomale e determinando in questo modo l’esecuzione automatica di azioni di recupero (*reboot* o *power-cycle* del sistema, riavvio di processi, ecc.).

In una tipica installazione, i *server* dell’*Event Logger*, del *Task Manager* e del *Monitor System* sono in esecuzione su ogni nodo della *farm*, i *server* del *Power Manager* e del *Process Controller* sono in esecuzione su un numero ridotto di PC, denominati *control-PC*, ciascuno dei quali controlla circa 200 nodi della *farm*, infine i *client* sono in esecuzione su uno o più PC dedicati al controllo e al monitor (*monitor-PC*).

2.4 Il “middleware” DIM

DIM è uno strato *software* di comunicazione di rete, costruito sul protocollo TCP e basato sul paradigma *client/server*, che fa uso di un meccanismo di *name server*, figura(2.8). I *server* mettono a disposizione dei *client* (“pubblicano”) sia servizi-dati (*data service*), sia servizi-comandi (*command service*) “registrandosi” presso il *name server*. I *client* “sottoscrivono” i servizi domandando al *name server* quali *server* forniscono i servizi richiesti, quindi contattando i *server* direttamente.

L’aggiornamento dei dati sul lato *client* può avvenire in 3 diverse modalità: ONCE_ONLY (il *client* riceve i dati una sola volta per poi disconnettersi dal *server*), TIMED (il *client* riceve i dati a intervalli regolari di tempo) oppure MONITORED (il *client* riceve i dati soltanto quando il *server* li aggiorna).

Il *client* può anche inviare un comando a un servizio-comandi per richiedere l’esecuzione di una procedura sul lato *server*.

⁴A tal fine, in caso di riavvio del *Task Manager*, un meccanismo che sfrutta la tavola dei processi del *kernel* Linux, permette alla nuova istanza del *Task Manager* di prendere il controllo di tutti i processi avviati dall’istanza precedente. Infine, in caso di terminazione inaspettata

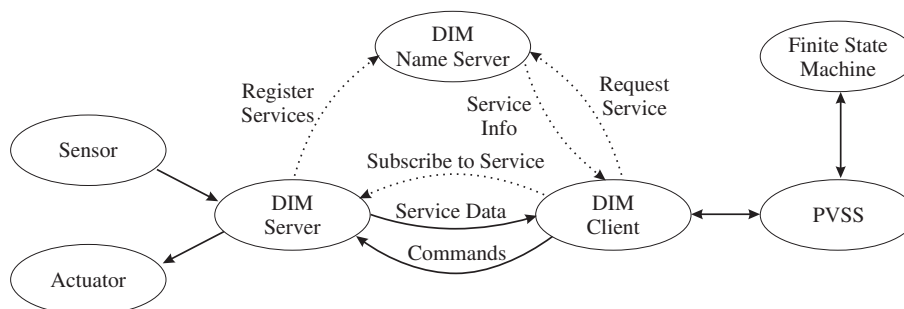


Figura 2.8: Diagramma del meccanismo di comunicazione di DIM.

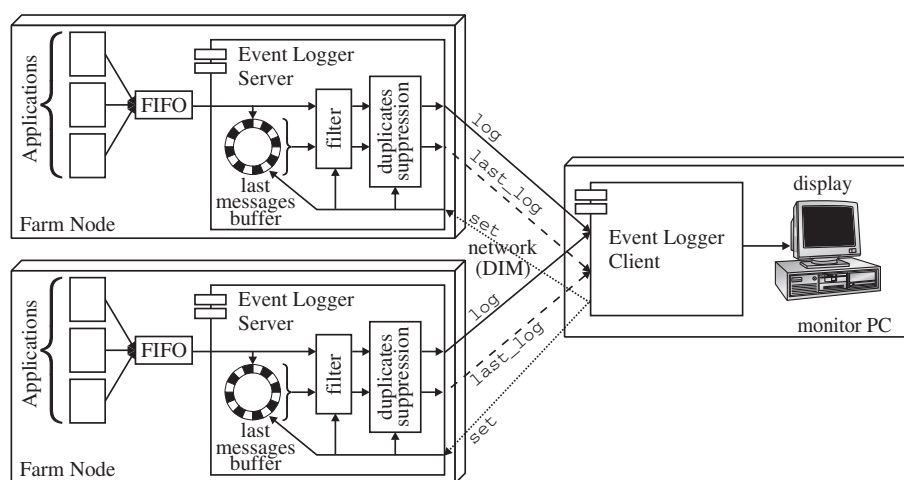


Figura 2.9: Diagramma delle componenti dell’*Event Logger*. Il *client* sottoscrive permanentemente il servizio DIM “log” mentre il servizio DIM “last_log” è acceduto dal *client* solo dopo un cambiamento nella configurazione del *server* tramite il comando “set”.

2.5 L'Event Logger

L'*Event Logger* [24] è uno strumento che raccoglie, gestisce (cioè mette insieme, duplica, filtra, sopprime i doppioni) e mostra su di una console centrale i messaggi diagnostici inviati da tutte le applicazioni in esecuzione sui nodi dell'EFF. L'*Event Logger* utilizza DIM per muovere i messaggi attraverso la rete e una FIFO POSIX.1 (nota anche come *named pipe*) [25][26], come memoria tampone (*buffer*) locale, per raccogliere i messaggi inviati dalle applicazioni in esecuzione sul nodo, figura(2.9).

L'*Event Logger* è uno strumento di utilità generale, non costruito su misura per una specifica applicazione; esso può persino raccogliere i messaggi inviati dalle applicazioni al loro *standard output stream* e *standard error stream*, redirezionando tali flussi di dati verso la FIFO, all'avvio dell'applicazione⁵.

dei processi del sistema FMC, essi sono automaticamente riavviati — o dal processo `init` di Linux o dal *Process Controller* del sistema FMC

⁵Il *Task Manager* mette a disposizione un'opzione nel comando d'avvio dei processi per

Quest'ultima caratteristica è particolarmente utile per intercettare i messaggi d'errore inviati dal *Dinamic Loader* di Linux allo *standard error stream* durante l'avvio delle applicazioni che fanno uso di *library* condivise (per esempio quando il *Dinamic Loader* non trova una particolare *library* condivisa). Senza la possibilità di redirezionare lo *standard error stream*, tali messaggi di errore non sarebbero recuperabili.

Poiché l'*Event Logger* utilizza DIM per muovere i messaggi sulla rete e DIM, a sua volta, si basa sul protocollo TCP, esso si avvantaggia dei meccanismi propri del protocollo TCP (come gli algoritmi di *slow start*, *congestion avoidance*, *fast retransmit* e *fast recovery* [27]) per evitare di contribuire ad accidentali congestioni della rete. Inoltre, l'*Event Logger* può completamente evitare la perdita di messaggi sulla rete (modalità di funzionamento *no-drop*) se tale caratteristica è ritenuta preferibile.

La FIFO locale nell'*Event Logger* agisce sia come meccanismo di comunicazione fra i processi (per trasferire i messaggi dalle applicazioni in esecuzione su un determinato nodo al *server* dell'*Event Logger* in esecuzione sullo stesso nodo) sia come memoria tampone (*buffer*) locale (per fare fronte a raffiche di messaggi e a stati di congestione temporanea della rete).

Come meccanismo di comunicazione fra processi, la FIFO può essere letta o scritta da molti processi concorrenti (anche di diversa discendenza), non richiede l'utilizzo di *semafori* o *mutex* da parte dei processi coinvolti, e garantisce l'integrità dei messaggi (atomicità delle operazioni di lettura e scrittura) a patto che la lunghezza dei messaggi sia inferiore del valore del parametro PIPE_BUF del *kernel*, correntemente impostato a 4 KiB (altrimenti risulta possibile la commistione di frammenti provenienti da messaggi diversi). Un'applicazione può perciò inviare messaggi all'*Event Logger* come se essa scrivesse su di uno *standard stream*, senza effettuare chiamate a primitive di arbitraggio del sistema operativo.

Come memoria tampone (*buffer*) locale, nei *kernel* Linux precedenti alla versione 2.6.11 la struttura dati di una FIFO è una lista circolare che occupa esattamente una pagina di memoria (4 KiB corrispondenti a 51 linee di un terminale standard 80×24); a partire dalla versione 2.6.11 del *kernel* Linux la struttura dati di una FIFO è diventata una struttura circolare di 16 pagine di memoria [28] (64 KiB corrispondenti a 819 linee di un terminale standard 80×24).

2.5.1 L'estrazione dei messaggi dalla FIFO

Il meccanismo di estrazione dei messaggi dalla FIFO, da parte del processo *server* dell'*Event Logger* è particolarmente ottimizzato e merita una descrizione dettagliata. Tale meccanismo si basa sulla **notificazione asincrona** della presenza di messaggi non letti nella FIFO, per mezzo del segnale SIGIO/SIGPOLL[29][26] (si veda la figura (2.10)).

A partire dal *kernel* Linux 2.5.20 la notificazione asincrona di I/O (*input/output*) mediante il segnale SIGIO/SIGPOLL è stata estesa alle *pipe* e alle FIFO (precedentemente era disponibile soltanto per terminali, pseudo-terminali e *socket*).

Impostando l'opzione O_ASYNC sul descrittore di lettura della *pipe* o della

attivare questo redirezionamento.

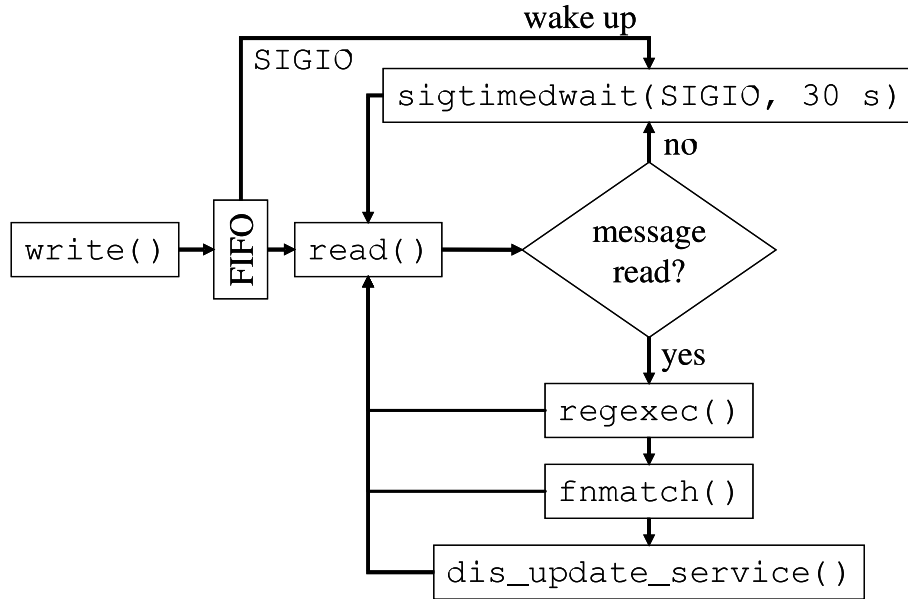


Figura 2.10: Diagramma di flusso del processo *server* dell'*Event Logger*, che estrae i messaggi dalla FIFO e li pubblica in un servizio DIM. Il meccanismo di estrazione dei messaggi converge verso un meccanismo di *interrupt* a basso rateo di messaggi e verso un meccanismo di *polling* ad alto rateo.

FIFO (mediante la *system call* `fcntl` [30]) un segnale asincrono (di *default* il segnale SIGIO che è sinonimo di SIGPOLL) è generato ogniqualvolta nuovi dati sono disponibili nella FIFO.

Come si può osservare nel diagramma di flusso della figura (2.10), quando sono disponibili nuovi messaggi nella FIFO il *kernel* del sistema operativo invia un segnale di SIGIO al processo *server* dell'*Event Logger*, il quale interrompe immediatamente lo stato di attesa procurato dalla *system call* `sigtimedwait` [31] e inizia un ciclo di lettura-pubblicazione dei messaggi (*polling*), realizzato dalla *system call* `read` e dalla DIM *library call* `dis_update_service`.

Tale ciclo di *polling* procede continuamente alla massima velocità consentita dal sistema, senza alcun tipo di attesa, finché ci sono messaggi nella FIFO (cioè finché la *system call* `read` restituisce un valore diverso da zero).

Quando non ci sono più messaggi nella fifo (e la *system call* `read` restituisce il valore zero) è invocata nuovamente la *system call* `sigtimedwait` che pone nuovamente il processo nello stato di attesa della ricezione di un segnale.

Il processo rimane perciò in tale stato di attesa (a meno di un ciclo ogni 30 secondi, allo scadere del *timeout* della chiamata `sigtimedwait`) finché nuovi messaggi vengono inseriti nella FIFO e il sistema operativo invia nuovamente un segnale di SIGIO al processo il quale, a sua volta, riavvia il *polling*.

In questo modo, quando il rateo di messaggi è basso, la logica del processo di lettura converge verso un meccanismo di *interrupt*: il processo rimane in stato di attesa e si risveglia soltanto ogniqualvolta arriva un segnale di SIGIO che notifica la presenza di un nuovo messaggio nella FIFO. Con tale logica il processo non spreca inutilmente risorse di CPU per sondare (*polling*) se ci sono

nuovi messaggi nella FIFO quando essa rimane vuota per la maggior parte del tempo.

Quando invece il rateo di messaggi è elevato (raffiche di messaggi) la logica del processo di lettura converge verso un meccanismo di *polling*: il processo legge e pubblica i messaggi alla massima velocità consentita dal sistema, senza stati di attesa e senza dovere reagire a ogni ricezione di un segnale di SIGIO⁶.

2.5.2 Modalità di funzionamento “No-drop” e “Congestion-Proof”

L'*Event Logger* del sistema FMC può operare sia in modalità *no-drop* sia in modalità *congestion-proof*.

Se si verifica la condizione in cui la FIFO è completamente piena (a causa di una congestione temporanea della rete o per l'arrivo di una raffica di messaggi a un rateo superiore al rateo del loro smistamento), allora un tentativo di scrittura nella FIFO produce effetti diversi nelle due modalità di funzionamento:

- in modalità *no-drop* il tentativo blocca il processo scrivente finché un messaggio non è estratto dalla FIFO, liberando lo spazio necessario per inserire il nuovo messaggio;
- in modalità *congestion-proof* il tentativo restituisce l'errore EAGAIN al processo scrivente senza bloccarlo e senza inserire il messaggio nella FIFO (il messaggio è perso).

La modalità *no-drop* può essere utile in fase di *debug*: nessun messaggio è perso ma, in condizioni critiche, le applicazioni potrebbero bloccarsi in attesa di poter inviare un messaggio.

La modalità *congestion-proof* è invece preferibile durante le normali operazioni di acquisizione dati: in caso di congestione della rete i messaggi sono persi ma le applicazioni non si bloccano e nessun traffico aggiuntivo di messaggi è iniettato nella rete già congestionata (eccetto le ri-trasmissioni TCP, le quali tuttavia sono drasticamente rallentate dal meccanismo TCP di controllo della congestione), in quanto i messaggi sono scartati nella FIFO, **prima** che essi raggiungano la rete.

Se l'applicazione apre direttamente la FIFO per scrivere, allora la modalità può essere scelta specificando o meno l'opzione O_NONBLOCK nella chiamata di sistema `open` utilizzata per accedere alla FIFO in scrittura. Se invece un'applicazione invia messaggi allo *standard output stream* o allo *standard error stream* e si utilizza il *Task Manager* per reindirizzare tali messaggi verso l'*Event Logger*, allora la modalità può essere scelta mediante un'opzione nel comando di avvio di un processo inviato al *Task Manager*.

2.5.3 Il filtro dei messaggi

Molto spesso l'eccesso di messaggi nasconde quelli di reale interesse per cui è spesso desiderabile poter filtrare il flusso dei messaggi. L'*Event Logger* premette

⁶In un meccanismo basato puramente sull'*interrupt*, se il rateo di interruzioni è eccessivamente elevato per la capacità di elaborazione del sistema, si determina uno stato di paralisi denominato stato di *interrupt live-lock*, nel quale il sistema utilizza tutte le proprie risorse per reagire agli *interrupt* e non ha più risorse disponibili per elaborare le informazioni che tali *interrupt* notificano

di filtrare tale flusso sia sul lato *server* sia sul lato *client*. La selezione dei messaggi può essere basata su:

- la severità dei messaggi, classificata in sei categorie (VERBOSE, DEBUG, INFO, WARN, ERROR e FATAL) e riconosciuta o da un'intestazione (*header*) convenzionale del messaggio o per mezzo di un metodo euristico basato sulle *extended regular expression* [33], le cui impostazioni possono essere modificate all'avvio del *server*;
- un *wildcard pattern* arbitrario [32];
- una *extended regular expression* arbitraria [33].

La configurazione del filtro dei messaggi (soglia di severità, *wildcard pattern* ed *extended regular expression*) può essere modificata durante il funzionamento dell'*Event Logger* senza interrompere il flusso dei messaggi.

2.5.4 La soppressione dei duplicati

Quando molti processi sono in esecuzione su una grande *farm* di calcolatori, una condizione anomala (per esempio un disco condiviso non accessibile) può attivare lo stesso messaggio d'errore (o la stessa sequenza di messaggi di errore) su tutti i nodi della *farm*. Se lo stesso messaggio è inviato più di una volta da ogni nodo della *farm*, il risultato è una tempesta di messaggi che differiscono tra loro, al più, per pochi dettagli (ad esempio il nome del nodo che li ha spediti o un numero seriale ecc.).

Per evitare tale condizione, l'*Event Logger* possiede un meccanismo di soppressione dei messaggi duplicati.

Per riconoscere i duplicati, l'*Event Logger* confronta gli ultimi n messaggi ricevuti (in questo modo sopprimendo i messaggi ripetuti anche se intramezzati da zero fino a $n-1$ messaggi differenti), dove il parametro n può essere modificato durante l'esecuzione.

Per decidere se due messaggi sono differenti oppure se uno è il duplicato dell'altro, possono essere utilizzati tre diversi criteri di confronto:

- *confronto esatto*: due messaggi sono considerati differenti se il confronto, lettera a lettera, dei due messaggi evidenzia almeno un carattere diverso.
- *numero di parole diverse*: due messaggi sono considerati diversi se il numero di parole diverse nei due messaggi supera una soglia prestabilita.
- *distanza di Levenshtein*⁷: due messaggi sono considerati diversi se la distanza di Levenshtein⁷ tra i due supera una soglia prestabilita.

Poiché l'intestazione dei messaggi — che contiene la data, l'ora e il nome del nodo mittente — è quasi sempre diversa tra un messaggio e l'altro e potrebbe perciò essere fuorviante al fine del riconoscimento dei duplicati, è conveniente che essa sia omessa nel confronto. A tal fine l'*Event Logger* permette di iniziare

⁷La distanza di Levenshtein (conosciuta anche come distanza di redazione, *edit distance*) [35] è una misura della similarità di due stringhe di caratteri ed è definita come il numero minimo di operazioni di redazione (cioè di cancellazione, inserimento o sostituzione di un singolo carattere) necessarie per trasformare una stringa nell'altra.

il confronto dopo un numero fissato di caratteri dall'inizio del messaggio oppure dopo che nel messaggio si è presentata una prefissata sequenza di caratteri (non necessariamente contigui).

Tutte le configurazioni dell'*Event Logger* che riguardano le soppressioni dei duplicati (il tipo di confronto, la soglia nel numero di parole diverse, la soglia nella distanza di Levenshtein, il numero di messaggi da confrontare, il numero di caratteri da ignorare e la sequenza di caratteri da ignorare) possono essere cambiati anche durante il funzionamento del *server* senza interrompere il flusso dei messaggi.

2.5.5 Il “buffer” degli ultimi messaggi

Quando il filtro dei messaggi ha impostazioni molto stringenti soltanto i messaggi più severi sono mostrati sulla console, mentre tutti gli altri messaggi sono scartati. Se succede qualcosa di grave potrebbe tuttavia essere utile poter consultare anche i messaggi di minor severità già scartati.

A questo scopo il *server* dell'*Event Logger* implementa una memoria tampone (*buffer*) che contiene gli ultimi m messaggi (prima del filtro e della soppressione dei duplicati, figura (2.9), dove il parametro m può essere modificato anche durante il funzionamento del *server* senza interrompere il flusso dei messaggi.

L'intero *buffer* può essere recuperato — filtrato e ridotto con le impostazioni correnti — attraverso il servizio DIM nominato `last_log`.

Mentre in condizioni normali il *client* dell'*Event Logger* sottoscrive, in modo MONITORED, soltanto il servizio DIM `log` che contiene esclusivamente l'ultimo messaggio, ogniqualvolta le impostazioni (di filtro e di soppressione dei duplicati) del *server* sono modificate, il *client* accede anche, in maniera ONCE_ONLY, al servizio DIM `last_log` per recuperare il *buffer* degli ultimi messaggi (si veda la figura 2.11).

Quando l'operatore di turno si rende conto — leggendo i messaggi più severi — della presenza di problemi gravi, egli può modificare le impostazioni di filtro e di soppressione dei duplicati del nodo in questione — rendendole più lasche — e in tal modo egli riceverà nuovamente gli ultimi m messaggi, questa volta filtrati con le impostazioni più lasche.

2.5.6 Prestazioni dell'Event Logger

La figura 2.12 mostra un grafico delle prestazioni ottenute usando un computer Dell PowerEdge SC 1450, dotato di due CPU intel Xeon a 64 bit (3.2 GHz di clock, 2048 KiB di cache), un chipset Intel E7520 (con bus di sistema a 800 MHz), 2 GiB di RAM e un controllore Gigabit Ethernet Intel 82541 GM. Il sistema operativo usato è Scientific Linux CERN SLC 4.4 (Berilium) con *kernel* 2.6.9, compilato per l'architettura x86_64.

I messaggi sono inviati dall'applicazione di test alla FIFO, quindi estratti dalla FIFO dal *server* dell'*Event Logger* e inviati a un *client* dell'*Event Logger* in esecuzione sullo stesso nodo, senza filtri o soppressione dei duplicati, utilizzando la modalità *no-drop* di funzionamento.

Come si può osservare dal grafico, il massimo rateo di *bit* aumenta al crescere delle dimensioni del messaggio — a causa dell'*over-head* di messaggio — fino a raggiungere circa 400 Mb/s.


```

Terminal
File Edit View Terminal Tabs Help
Nov24-170255[ERROR]lhbsrv: sendMsg: main(): 3 error error error error error er
Nov24-170256[FATAL]lhbsrv: sendMsg: main(): 4 fatal fatal fatal fatal fatal fa
Nov24-170300[ERROR]lhbsrv: sendMsg: main(): 8 error error error error error er
Nov24-170301[FATAL]lhbsrv: sendMsg: main(): 9 fatal fatal fatal fatal fatal fa
Nov24-170307[WARN] lhbsrv: logSrv(logger_1): printChangedSettings(): Filter set
tings changed: no Severity filter, no Regular Expression filter, no Wildcard Pat
tern filter.
----- old messages from: /HBTSEV/logger_1/log
Nov24-170254[WARN] lhbsrv: sendMsg: main(): 2 warning warning warning warning
Nov24-170255[ERROR]lhbsrv: sendMsg: main(): 3 error error error error error er
Nov24-170256[FATAL]lhbsrv: sendMsg: main(): 4 fatal fatal fatal fatal fatal fa
Nov24-170257[DEBUG]lhbsrv: sendMsg: main(): 5 debug debug debug debug debug de
Nov24-170258[INFO] lhbsrv: sendMsg: main(): 6 info info info info info info in
Nov24-170259[WARN] lhbsrv: sendMsg: main(): 7 warning warning warning warning
Nov24-170300[ERROR]lhbsrv: sendMsg: main(): 8 error error error error error er
Nov24-170301[FATAL]lhbsrv: sendMsg: main(): 9 fatal fatal fatal fatal fatal fa
Nov24-170302[DEBUG]lhbsrv: sendMsg: main(): 10 debug debug debug debug debug d
Nov24-170307[WARN] lhbsrv: logSrv(logger_1): printChangedSettings(): Filter set
tings changed: no Severity filter, no Regular Expression filter, no Wildcard Pat
tern filter.
----- 0 old messages filtered by client.
----- 0 duplicated old messages suppressed by client.
----- end old messages from: /HBTSEV/logger_1/log

```

Figura 2.11: Utilizzo del *buffer* degli ultimi messaggi. Inizialmente il filtro è impostato per lasciar passare soltanto i messaggi con severità di tipo ERROR e FATAL. Successivamente la configurazione del filtro è modificata per lasciar passare anche i messaggi di severità di tipo DEBUG, INFO e WARN. Si può osservare, tra le due righe blu, la riproposizione dei vecchi messaggi (ripescati nel *buffer* degli ultimi messaggi), che include anche quelli di bassa severità.

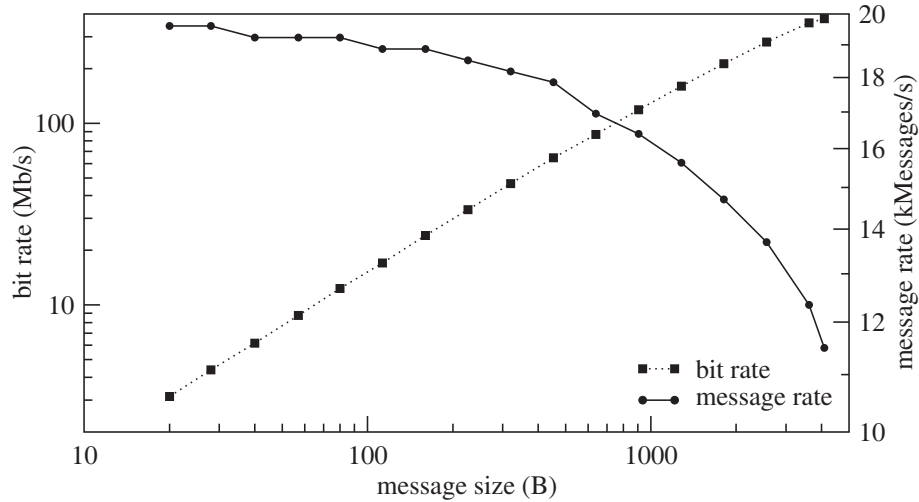


Figura 2.12: Grafico del massimo rateo di bit e messaggi dell'*Event Logger* in funzione delle dimensioni del messaggio. Ogni punto è la media di 10 misure.

Il massimo rateo di messaggi, invece, diminuisce con l'aumentare della lunghezza del messaggio, ma con pendenza che rimane contenuta per messaggi più brevi di 600 caratteri: in questo intervallo esso si mantiene al di sopra di $1.7 \cdot 10^4$ messaggi per secondo.

2.6 Il Power Manager

Il *Power Manager* [36] è uno strumento per accendere, spegnere, effettuare un ciclo di alimentazione (*power-cycle*) — cioè uno spegnimento seguito da una riaccensione — e per mostrare, su di una console centrale, lo stato di alimentazione (acceso/spento) e lo stato delle informazioni dei sensori accessibili mediante il bus I²C (ad esempio le temperature, la velocità delle ventole, le tensioni di alimentazione, le correnti assorbite, ecc.) di ogni nodo.

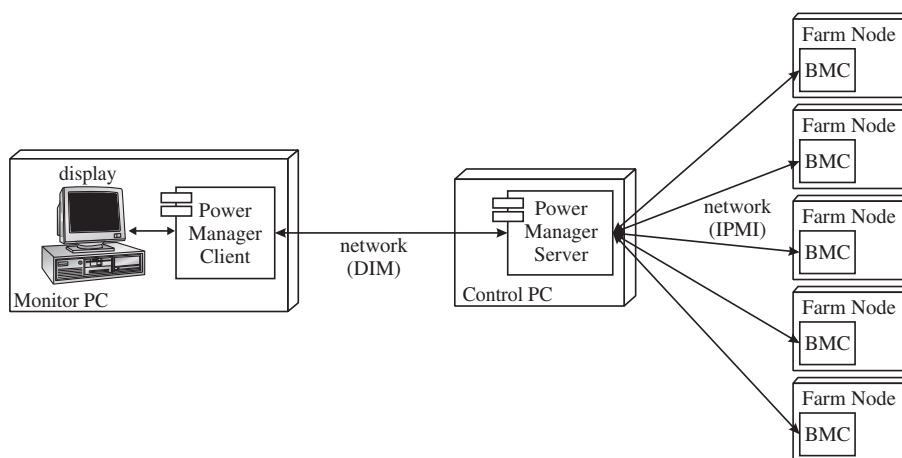
Nell'installazione di LHCb, il *server* del *Power Manager* è in esecuzione su un numero ridotto di PC della *farm*, denominati *control-PC*, ciascuno dei quali sovrintende a circa 200 nodi, figura (2.13). L'esperimento CMS prevede invece di utilizzare un'unica istanza del *Power Manager* per sovrintendere a circa 5000 nodi (è già stato effettuato con successo un test con 1200 nodi).

Per comunicare con i *client* (che inviano comandi e leggono lo stato), il *server Power Manager* usa DIM come *server*, mentre per comunicare con i nodi esso usa IPMI (Intelligent Platform Management Interface) come *client* (i *server* IPMI sono realizzati dal *firmware* in esecuzione sulle *motherboard* dei nodi).

IPMI [37] è un'interfaccia standardizzata, astratta, basata sullo scambio di messaggi, verso l'*Intelligent Platform Management hardware* la quale definisce i *record* per descrivere i dispositivi di gestione della piattaforma e le loro caratteristiche.

IPMI ha due tipi di interfacce:

- l'interfaccia **KCS** (Keyboard Controller Style) conosciuta anche come

Figura 2.13: Schema dell'installazione del *Power Manager*.

Open Interface, che è un'interfaccia locale (interfaccia verso il sistema operativo ospite), senza autenticazione che può essere acceduta attraverso il *software* Linux OpenIPMI [38] in esecuzione sul nodo medesimo e che dunque non può essere usata per accendere un PC o effettuare un *power-cycle* di un PC bloccato;

- l'interfaccia **LAN**, che è un'interfaccia di rete, basata su sessione, autenticata, progettata per essere sempre disponibile — anche quando il nodo è spento o il sistema operativo è bloccato o inattivo — e perciò è implementata in *hardware* e indipendente dal sistema operativo.

L'interfaccia LAN di IPMI è implementata mediante un *microcontroller* (denominato BMC, *Baseboard Management Controller*), incastonato nella scheda madre, che è alimentato anche quando il PC è spento. Lo *hardware* dell'interfaccia di rete redirige al controller BMC i *frame* Ethernet contenenti datagrammi indirizzati verso la porta UDP numero 623. L'interfaccia LAN di IPMI può essere perciò raggiunta attraverso lo stesso connettore Ethernet usato per l'interfaccia di rete standard, sia utilizzando lo stesso indirizzo IP, sia utilizzando un indirizzo IP dedicato.

IPMI può essere configurato per mezzo di una *utility* di configurazione all'avvio del PC e può fare uso del protocollo DHCP [39] per impostare i parametri di rete. L'interfaccia LAN di IPMI non richiede l'uso di alcun *software* sul lato *server*.

Il *server Power Manager* è un *server* DIM che connette DIM all'interfaccia LAN di IPMI e sopperisce a diverse limitazioni della versione 1.5 di IPMI.

Il *server Power Manager*, sfruttando il *multi-thread*, è in grado di interrogare **in parallelo** tutti i nodi controllati, minimizzando l'effetto dei lunghi tempi di risposta di IPMI (che possono essere maggiori di 0.7 s) e dei lunghissimi tempi di risposta in caso di *time-out* (circa 16 s per nodi disconnessi). Tra due successivi aggiornamenti periodici dello stato dei sensori di IPMI, il *server Power Manager* mantiene memorizzati i valori rilevati, insieme alla data e al tempo di lettura e può così rispondere immediatamente alla richiesta di un *client*.

Arbitrando i comandi IPMI inviati allo stesso singolo nodo, il *Power Manager* è anche in grado di sopperire alla capacità di IPMI di elaborare soltanto **un accesso alla volta** (se il controller BMC sta elaborando un accesso non è in grado di gestirne o accodarne altri: il secondo accesso semplicemente fallisce e non è elaborato).

Una sovrapposizione di accessi a IPMI può avvenire, ad esempio, se si invia un comando di spegnimento mentre il BMC sta accedendo alle informazioni dei sensori per un aggiornamento periodico. Se il comando di spegnimento è inviato a tutta la *farm* di 2000 PC, l'esito è catastrofico: ci si ritrova una parte della *farm* spenta e la restante parte ancora accesa.

In caso di accessi IPMI sovrapposti sullo stesso nodo, il *Power Manager* gestisce la situazione **accodando i comandi** (accensione, spegnimento, *power-cycle*) per eseguirli, uno alla volta, nello stesso ordine temporale in cui sono stati inviati e **annullando gli aggiornamenti periodici dei sensori** (eventualmente attendendo la terminazione dell'aggiornamento già in esecuzione) per evitare un accumulazione indefinita di *thread* in caso di un'interfaccia IPMI che non risponde.

2.7 Il Task Manager

Il *Task Manager* [40] è uno strumento del sistema FMC che consente, tramite una console centrale, di:

- **avviare un processo** in uno o più nodi della *farm*;
- ottenere la **lista dei processi** avviati in uno o più nodi della *farm*;
- **inviare un segnale** a uno o più processi in esecuzione nella *farm*;
- **terminare uno o più processi** in esecuzione nella *farm*.

gestendo sia lo *scheduler time-sharing* sia gli *scheduler real-time*, configurando la *CPU affinity* (ovvero l'assegnazione di un processo a una ben definita CPU nei sistemi multiprocessore), la priorità, le variabili di ambiente, il proprietario del processo e, se richiesto, reindirigendo lo *standard output stream* e/o lo *standard error stream* del processo verso l'*Event Logger*. Il *Task Manager* è composto da un *server* DIM, che deve essere in esecuzione su ciascun nodo della *farm*, e da alcuni *client* DIM, che inviano comandi o recuperano informazioni dal *server*.

2.7.1 L'UTGID (User assigned Thread Group Identifier)

Per poter controllare, inviare segnali o arrestare i processi in esecuzione, il *server Task Manager* deve essere in grado di **tenere traccia** dei processi che esso stesso ha avviato.

Il sistema operativo Linux identifica i processi (ai fini dello *scheduling*) utilizzando un numero intero, denominato TGID (Thread Group Identifier, la nuova denominazione, nell'epoca del *multi-thread*, del precedente PID, Process Identifier). Associato al TGID c'è la stringa di caratteri COMMAND, che è il nome dell'immagine eseguibile del processo nel *file system*. Nessuno di questi due identificatori, presi singolarmente, è tuttavia utilizzabile per tenere traccia del processo: il TGID è un numero intero assegnato sequenzialmente, così che allo stesso

processo è assegnato un TGID diverso se esso è riavviato; la stringa `COMMAND`, d'altro canto, è la stessa per due differenti istanze dello stesso processo.

Per tenere traccia dei processi, il *Task Manager* assegna a ciascun processo avviato un ulteriore identificativo, una stringa di caratteri descrittiva, denominata `UTGID` (User assigned Thread Group Identifier). L'`UTGID` — possibilmente ma non necessariamente composta dalla stringa `COMMAND` seguita da un carattere di sottolineatura (*underscore*) e un contatore di istanza — deve essere unico su un nodo: una stesso `UTGID` non può essere usato per più di un processo sullo stesso PC. Nei processi *multi-thread*, è assegnato lo stesso `UTGID` a tutti i *thread* dello stesso processo ma solo il *thread group* (il processo) è elencato nei servizi `list` e `longList` e i segnali sono inviati soltanto al *thread group*.

L'`UTGID` è memorizzato nell'*array* delle **variabili di ambiente** [41], passata al processo nello *stack* iniziale, insieme all'*array* degli argomenti della linea di comando. L'`UTGID` rimane perciò *incollato* al processo nella *process table* del sistema operativo e sopravvive senza problema persino al riavvio del *server* del *Task Manager*⁸.

L'`UTGID`, come del resto tutte le altre variabili di ambiente, può essere impostato anche dall'interno del processo stesso ma diviene accessibile all'esterno, mediante il `/proc file system`⁹ [42], soltanto dopo che è stata eseguita la *system call* `execve` [43].

Poiché la creazione di un nuovo processo, nel sistema operativo Linux, richiede, in sequenza, la *system call* `fork` [44] — che duplica il processo corrente (con la medesima immagine eseguibile) — e la *system call* `execve` [43] — che sostituisce l'immagine eseguibile corrente del processo con una nuova immagine eseguibile — segue che l'`UTGID` deve essere impostato dopo la *system call* `fork` e prima della *system call* `execve`.

2.7.2 L'avvio di un processo

Ogni *server* del *Task Manager* (c'è un *server* del *Task Manager* in esecuzione su ogni nodo della *farm*) “pubblica” sul *name server* DIM il comando “`start`” per avviare processi sullo stesso nodo sul quale esso è in esecuzione.

Per avviare un nuovo processo su uno o più nodi della *farm*, il *client* del *Task Manager* contatta il comando `start` presso uno o più *server Task Manager*.

Il *server*, a sua volta, ricevendo il comando `start`, invoca nell'ordine le chiamate di sistema `fork` ed `execve`, eseguendo le seguenti impostazioni tra le due chiamate:

1. La *directory* di lavoro corrente (*current working directory*¹⁰) è impostata al percorso (*path*) desiderato (chiamata di sistema `chdir` [45]).
2. Il processo può essere avviato ereditando le variabili d'ambiente [41] del *Task Manager* (opzione utile se si hanno molte variabili di ambiente che devono essere definite in comune a molti processi) o può essere avviato

⁸In caso di riavvio del *Task Manager*, la nuova istanza del *Task Manager* ritrova, nella *process table* del sistema operativo, gli `UTGID` dei processi avviati dalla precedente istanza, e può, in tal modo, prenderne il controllo.

⁹Nello *pseudo-file* `/proc/(tgid)/environ`.

¹⁰In essa sono cercati dal processo i nomi di *file* privi di percorso ed è considerata dal processo come punto di partenza dei percorsi relativi.

senza variabili d'ambiente predefinite. In quest'ultimo caso è eseguita la *library call* `clearenv` [46].

3. Per mezzo della *library call* `putenv` [47] può essere aggiunto, individualmente a ogni processo avviato, un numero arbitrario di variabili di ambiente.
4. Tramite la *library call* `setenv` [48] è impostata la variabile d'ambiente `UTGID`.
5. Tutti i *file descriptor* aperti dal processo sono chiusi (*system call* `close` [49]). Il *file descriptor* standard `STDIN_FILENO` è riaperto sul percorso `/dev/null` per distaccare lo *standard input stream*. I *file descriptor* standard `STDOUT_FILENO` e `STDERR_FILENO` possono essere entrambi riaperti sul percorso `/dev/null` (per scartare tutto ciò che il processo scrive sullo *standard output stream* e sullo *standard error stream*) oppure, se richiesto, possono essere entrambi riaperti sulla FIFO dell'*Event Logger* (per redirezionare sull'*Event Logger* lo *standard output stream* e lo *standard error stream* del processo) (chiamate di sistema `open` [50] e `dup2` [51]).
6. Se richiesto è impostata l'*affinity mask* processo-CPU (chiamata di sistema `sched_setaffinity` [52]) per legare un processo a una ben definita CPU e così migliorare le prestazioni del processo su macchine multi-processore (evitando il più possibile i *context switch* e il *popout* dei registri).
7. È impostata la *scheduling policy* richiesta per l'esecuzione del processo. Il *Task Manager* consente la scelta fra tre diverse *scheduling policy*:
 - `SCHED_OTHER`, lo scheduler **time-sharing** predefinito di Linux, che garantisce il miglior *fair-sharing* fra i processi ma che non può garantire un tempo di risposta breve e deterministico (cioè con piccole fluttuazioni) ai processi che ne necessitano. I processi gestiti da questo scheduler hanno priorità dinamica, basata sul *nice level* (compreso nell'intervallo $-20 \dots +19$, dove il valore -20 corrisponde allo *scheduling* più favorevole). La priorità è aumentata in ogni quanto temporale nel quale il processo è pronto per essere eseguito ma l'esecuzione è impedita dallo scheduler. La priorità è poi riportata al livello più basso dopo che al processo è stata assegnata la CPU in un quanto temporale. Il *nice level* modifica la lunghezza del quanto temporale di base.
 - `SCHED_FIFO`, la scheduling policy **real-time** di tipo **FIFO** (first in first out). La priorità dei processi gestiti dallo scheduler FIFO è fissa (compresa nell'intervallo $1 \dots 99$, dove un numero più alto corrisponde a uno *scheduling* più favorevole) e un processo in esecuzione può essere interrotto soltanto da un processo *real-time* con priorità più elevata. In questo modo, tra i processi aventi una data priorità, il primo a essere eseguito è portato a termine per primo e senza interruzioni da parte di altri processi di pari priorità; gli altri processi di pari priorità sono eseguiti soltanto dopo che tale processo è terminato, oppure se tale processo si pone in stato di attesa di un I/O di dati.

- **SCHED_RR**, la scheduling policy **real-time** di tipo **round-robin**. La priorità dei processi gestiti dallo *scheduler round-robin* è fissa (anch'essa compresa nell'intervallo 1 ... 99, dove un numero più alto corrisponde a uno *scheduling* più favorevole) ma — nel caso esistano più processi con la stessa priorità — ogni processo è eseguito soltanto per un fissato quanto temporale, dopo il quale la CPU è ceduta a un altro processo di pari priorità. In questo modo, in assenza di processi di priorità più alta, la CPU è equamente suddivisa tra i processi di una data priorità.

Se si sceglie uno *scheduler real-time* è assegnata al processo anche la *static priority* (conosciuta anche come *real-time priority*). La scelta dello *scheduler* e l'impostazione della *static priority* sono effettuate mediante la chiamata di sistema `sched_setscheduler` [53].

8. Se il processo è gestito con la *scheduling policy time-sharing* predefinita (`SCHED_OTHER`) è impostato anche il *nice level* mediante la chiamata di sistema `setpriority` [54].
9. Lo *user identifier* (UID) del processo è impostato al codice identificativo dell'utente che si vuole divenga proprietario del processo (chiamata di sistema `setuid` [55]).
10. Il *group identifier* (GID) del processo è impostato al codice identificativo del gruppo che si vuole divenga proprietario del processo (chiamata di sistema `setgid` [56]).

Se il processo è eseguito come *demone* (*daemon*) sono necessarie due ulteriori impostazioni:

11. La *user file-creation mask* (`umask`) del processo¹¹ è azzerata (chiamata di sistema `umask` [57]).
12. È creata una nuova sessione di processi (*process group*) e il processo è impostato come *process group leader* di tale sessione (chiamata di sistema `setsid` [58]).

2.7.3 L'invio di un segnale a un processo

Un segnale [29] è inviato a un processo, tipicamente, o per terminarlo oppure per indurlo a rileggere uno o più *file* di configurazione.

Nel sistema operativo Linux i segnali sono normalmente inviati ai processi per mezzo della chiamata di sistema `kill` [59]¹².

Ogni *server* del *Task Manager* “pubblica” sul *name server* DIM il comando “`kill`” per inviare un segnale a uno o più processi in esecuzione sullo stesso nodo sul quale esso stesso è in esecuzione. Il comando `kill` accetta due argomenti:

¹¹La `umask` modifica la maschera dei permessi di accesso dei nuovi file creati, la quale è impostata all'OR bit-a-bit tra la maschera dei permessi specificata nella chiamata di sistema `open` e la negazione bit-a-bit della `umask`.

¹²Tuttavia anche altre chiamate, come `pthread_kill`, `sigqueue` o `alarm` consentono di inviare un segnale a un processo.

- L'UTGID del processo al quale si vuole inviare il segnale (argomento obbligatorio). Per inviare un segnale a più di un processo con un singolo comando è anche consentito di specificare un insieme di UTGID mediante un unico *wildcard pattern* [32].
- Il segnale da inviare [29] (argomento facoltativo). Se tale segnale non è specificato, è inviato il segnale predefinito SIGTERM.

Per inviare un segnale a uno o più processi in esecuzione su di uno o più nodi della *farm*, il *client* del *Task Manager* contatta il comando `kill` presso uno o più *server Task Manager*, specificando il segnale da inviare e l'UTGID (o il *wildcard pattern* di UTGID).

Il *client* del *Task Manager*, per inviare un segnale a processi in esecuzione sulla *farm*, accetta quindi 3 argomenti: (a) lo *hostname* del nodo (o un *wildcard pattern* di *hostname*); (b) l'UTGID del processo al quale si vuole inviare il segnale (o un *wildcard pattern* di UTGID); (c) il segnale da inviare.

2.7.4 La terminazione di un processo

Il comando DIM “`stop`”, pubblicato dal *server* del *Task Manager*, è stato progettato per terminare i processi in esecuzione sulla *farm* nella maniera più efficace.

Ricevendo il comando `stop`, il *server* del *Task Manager* invia un segnale al processo (il segnale SIGTERM [29] è predefinito ma qualunque altro segnale può essere specificato in alternativa) e simultaneamente programma l'esecuzione ritardata (in un nuovo *thread*) della funzione `finishOffPs`.

La funzione `finishOffPs`, a sua volta, controlla se il processo è già terminato; in caso contrario invia un secondo segnale al processo — questa volta il segnale SIGKILL, che non può essere gestito, bloccato o ignorato dal processo.

In questo modo al processo è data la possibilità di terminare “regolarmente” alla ricezione del primo segnale — il segnale SIGTERM o il segnale specificato in alternativa — che può essere gestito dal processo per effettuare le operazioni necessarie per una terminazione pulita (per esempio il *flush* dell'I/O, il distacco o la rimozione delle strutture utilizzate per l'*inter-process communication*, ecc.); tuttavia, se per qualche motivo tale segnale non ottiene la terminazione del processo, dopo un certo *timeout*, che può essere impostato di volta in volta, il processo è terminato “drasticamente” dal segnale SIGKILL.

Il *client* del *Task Manager*, per terminare processi in esecuzione sulla *farm*, accetta quindi 4 argomenti: (a) lo *hostname* del nodo (o un *wildcard pattern* di *hostname*); (b) l'UTGID del processo che si vuole terminare (o un *wildcard pattern* di UTGID); (c) il segnale da inviare come primo segnale; (d) il *timeout* (in secondi) da attendere prima di inviare il secondo segnale (SIGKILL).

2.7.5 La lista dei processi in esecuzione

Assieme ai tre comandi descritti sopra il *server* del *Task Manager* “registra” inoltre sul *name server* DIM i servizi `list` e `longList`, che consente di ottenere una lista aggiornata dei processi in esecuzione a cui è assegnato un UTGID definito.


```

Terminal
File Edit View Terminal Tabs Help
-> tm1 -m lhbcn2 galli@lhbsrv 17:38

DIM_DNS_NODE: "lhbsrv.bo.infn.it" (from DIM_DNS_NODE environment variable).
Node patterns: "lhbcn2".
UTGID pattern: "*"
Node pattern: "lhbcn2"
Node: "lhbcn2". 19 process(es) have UTGID matching pattern: "*":
#  TGID  CMD          UTGID          STATUS
0  23682  logSrv        defaultLogSrv  running
1  23688  tmSrv         tmSrv_u        running
2  23691  fsSrv         fsSrv_u        running
3  23696  smartSrv      smartSrv_u     running
4  23702  tcpipSrv      tcpipSrv_u     running
5  23709  psSrv         psSrv_u        running
6  23712  oldPsSrv      oldPsSrv_u     running
7  23715  nifSrv        nifSrv_u       running
8  23716  memSrv        memSrv_u       running
9  23717  irqSrv        irqSrv_u       running
10 23722  oldIrqSrv     oldIrqSrv_u    running
11 23723  cpustatSrv    cpustatSrv_u   running
12 23730  oldCpustatSrv oldCpustatSrv_u running
13 23731  cpuinfoSrv    cpuinfoSrv_u   running
14 23732  oldCpuinfoSrv oldCpuinfoSrv_u running
15 23741  coalSrv       coalSrv_u       running
16 23748  oldCoalSrv    oldCoalSrv_u    running
17 24304  divisionByZero divisionByZero_0 terminated on signal 8 (SIGFPE) at Jan18-173506
18 24315  counter       counter_0       terminated on signal 15 (SIGTERM) at Jan18-173540
-> galli@lhbsrv 17:40

```

Figura 2.14: La lista dei processi in esecuzione avviati dal *Task Manager* (servizio *longList*).

Il servizio *list* riporta soltanto la lista degli UTGID, mentre il servizio *longList* riporta: TGID, COMMAND, UTGID, data e ora dell'ultima rilevazione e stato ("running" oppure "terminated", specificando nel secondo caso la causa della terminazione), si veda la figura (2.14).

La lista è ottenuta dal *server* del *Task Manager* accedendo a tutti gli *pseudo-file* `/proc/<TGID>/environ` nel *proc file system* e cercando in essi la variabile UTGID.

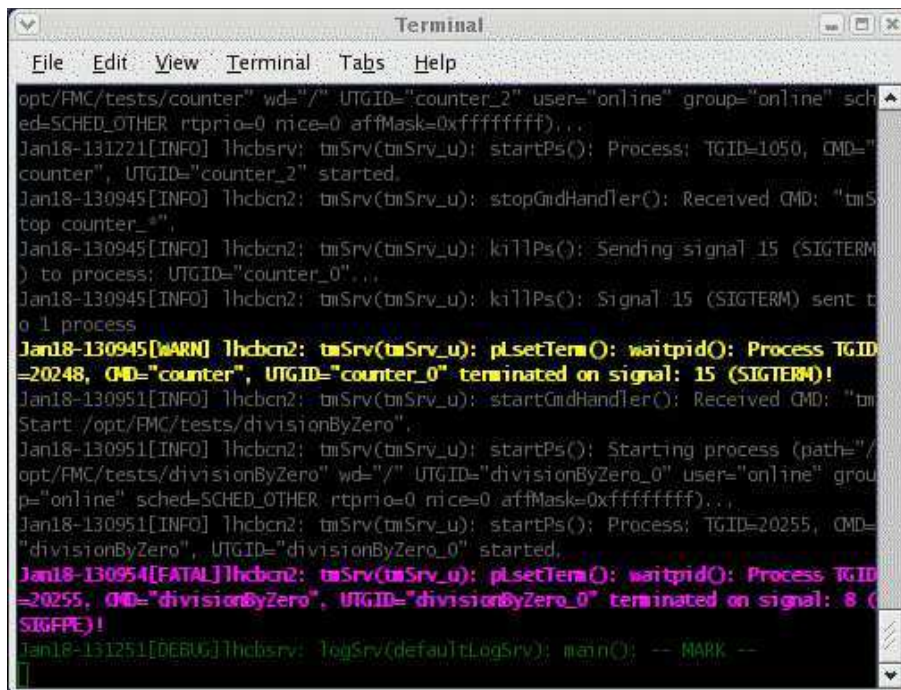
La lista pubblicata dal *server Task Manager* è aggiornata con le seguenti tre diverse modalità:

- periodicamente (il periodo predefinito è pari a 30 s ma può essere modificato);
- subito dopo che un comando del *Task Manager* (*start*, *kill* o *stop*) è stato eseguito;
- in seguito alla ricezione del segnale asincrono SIGCHLD, che segnala al *Task Manager* la terminazione (anche inaspettata) di un processo da esso stesso avviato, come descritto nel seguente paragrafo 2.7.6.

2.7.6 La reazione del Task Manager alla terminazione dei processi

Quando un processo avviato dal *Task Manager* termina — sia in seguito a un comando, sia inaspettatamente¹³ — esso permane nella lista dei processi

¹³Per esempio a causa di un *bug*, di un'eccezione non gestita o di un *memory leak* che produce, dopo un certo tempo, uno stato di memoria piena.



```

opt/FMC/tests/counter" wd="/" UTGID="counter_2" user="online" group="online" sched=
ed=SCHED_OTHER rtprio=0 nice=0 affMask=0xffffffff),...
Jan18-131221[INFO] Thcbn2: tmSrv(tmSrv_u): startPs(): Process: TGID=1050, CMD="
counter", UTGID="counter_2" started.
Jan18-130945[INFO] Thcbn2: tmSrv(tmSrv_u): stopCmdHandler(): Received CMD: "tmS
top counter_*",
Jan18-130945[INFO] Thcbn2: tmSrv(tmSrv_u): killPs(): Sending signal 15 (SIGTERM
) to process: UTGID="counter_0"...
Jan18-130945[INFO] Thcbn2: tmSrv(tmSrv_u): killPs(): Signal 15 (SIGTERM) sent t
o 1 process
Jan18-130945[WARN] Thcbn2: tmSrv(tmSrv_u): pLsetTerm(): waitpid(): Process TGID
=20248, CMD="counter", UTGID="counter_0" terminated on signal: 15 (SIGTERM)!
Jan18-130951[INFO] Thcbn2: tmSrv(tmSrv_u): startCmdHandler(): Received CMD: "tm
Start /opt/FMC/tests/divisionByZero",
Jan18-130951[INFO] Thcbn2: tmSrv(tmSrv_u): startPs(): Starting process (path="/
opt/FMC/tests/divisionByZero" wd="/" UTGID="divisionByZero_0" user="online" grou
p="online" sched=SCHED_OTHER rtprio=0 nice=0 affMask=0xffffffff),...
Jan18-130951[INFO] Thcbn2: tmSrv(tmSrv_u): startPs(): Process: TGID=20255, CMD=
"divisionByZero", UTGID="divisionByZero_0" started.
Jan18-130954[FATAL] Thcbn2: tmSrv(tmSrv_u): pLsetTerm(): waitpid(): Process TGID
=20255, CMD="divisionByZero", UTGID="divisionByZero_0" terminated on signal: 8 (
SIGFPE)!
Jan18-131251[DEBUG] Thcbn2: logSrv(defaultLogSrv): main(): -- MARK --

```

Figura 2.15: Messaggi inviati dal *Task Manager* all'*Event Logger* in seguito alla terminazione dei processi. Si osservi che il processo con UTGID "counter_0" è terminato in seguito a un segnale SIGTERM inviato dal *Task Manager* medesimo, mentre il processo con UTGID "divisionByZero_0" è terminata in seguito all'eccezione SIGFPE (*floating point exception*) prodotta dall'esecuzione di una divisione per zero.

del sistema operativo Linux — classificato come processo *zombie* — mentre il segnale asincrono SIGCHLD è inviato al processo genitore (cioè, in questo caso, al *server* del *Task Manager*), per segnalargli la terminazione del figlio.

Il *server* del *Task Manager* mantiene bloccato il segnale SIGCHLD (mediante la *system call* `pthread_sigmask` [60]) e si pone in stato di attesa di segnali bloccati (*system call* `sigtimedwait` [31]). Alla ricezione del segnale SIGCHLD, il *server* del *Task Manager* reagisce:

- invocando la *system call* `waitpid` [61], per **liberare** tutte le risorse utilizzate dal processo, **espungere** il processo dalla lista dei processi e **raccolgere** le informazioni sulla causa della terminazione, che può essere:
 - a) un *segnale di eccezione* (sincrono): SIGILL (illegal instruction), SIGBUS (bad memory access), SIGSEGV (invalid memory reference), SIGFPE (floating point exception), ecc.;
 - b) un *segnale asincrono*: SIGINT (interrupt from keyboard), SIGTERM (termination signal), SIGALRM (timer signal), SIGKILL (kill signal), ecc.;
 - c) una *terminazione normale*, ovvero un'istruzione **return** nella funzione **main** del programma oppure una chiamata alla *system call* `_exit` [62] o alla *library call* `exit` [63];
- inviando un **messaggio** all'*Event Logger* che specifica il processo (TGID, COMMAND e UTGID) e, in alternativa, o lo stato di terminazione (*exit status*), nel caso (c), oppure il numero e il nome del segnale che ha causato la terminazione, nei casi (a) e (b) (si veda la figura (2.15)).
- **aggiornando** immediatamente le **liste** dei processi “pubblicate” mediante DIM (`list` e `longList`) per consentire al *Process Controller*, se richiesto, di riavviare immediatamente il processo terminato.

2.7.7 Le prestazioni del Task Manager

La figura 2.16 mostra le prestazioni del *Task Manager* nell'avvio e nella terminazione di un gruppo di processi sul medesimo nodo della *farm*.

L'apparato di prova è il medesimo utilizzato per misurare le prestazioni dell'*Event Logger* (si veda il paragrafo 2.5.6 a pagina 49).

Il tempo di avvio è definito come l'intervallo di tempo intercorrente tra l'istante in cui il *client* impartisce il comando di avvio del gruppo di processi e l'istante in cui l'*Event Logger* riceve il primo messaggio inviato dall'ultimo processo avviato.

Il tempo di terminazione è definito come l'intervallo di tempo intercorrente tra l'istante in cui il *client* impartisce il comando di terminazione del gruppo di processi e l'istante in cui si conclude (sul *server*) la *system call* `waitpid` relativa all'ultimo processo terminato.

L'avvio di un singolo processo richiede mediamente un intervallo di tempo pari a 60–90 ms, valore che cresce lievemente con il numero di processi già in esecuzione sullo stesso nodo (probabilmente per il controllo di unicità dell'UTGID); la terminazione di un gruppo di processi richiede invece 0.4–0.5 s, quasi indipendentemente dalla dimensione del gruppo.

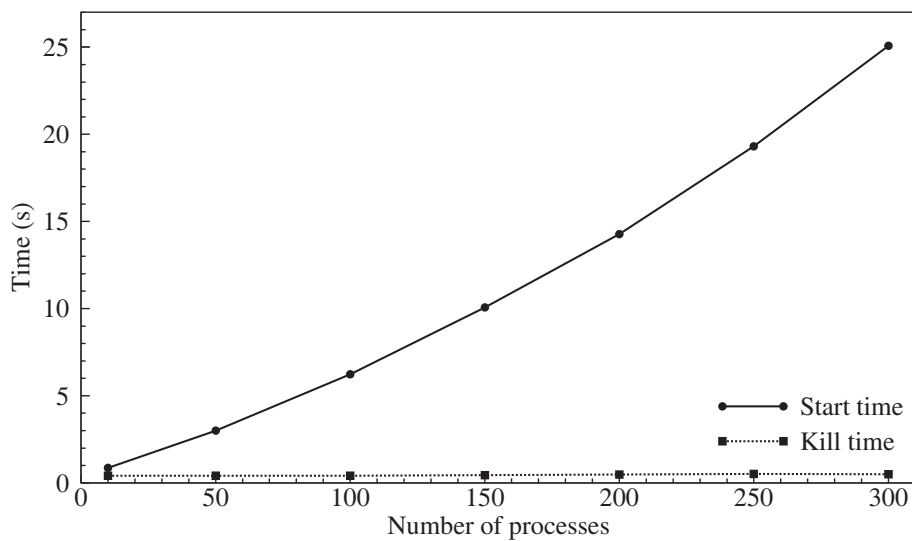


Figura 2.16: Grafico del tempo impiegato dal *Task Manager* per avviare e terminare un gruppo di processi in funzione della dimensione del gruppo. Ogni punto del grafico è il valor medio di 10 misure.

2.8 Il Process Controller

Il *Process Controller* [64] ha il compito di garantire il mantenimento in esecuzione di una lista di applicazioni sui nodi della *farm*. Esso adempie tale compito interagendo con i *server* del *Task Manager* in esecuzione sui nodi, per ottenere la notifica della terminazione dei processi e, se richiesto, per riavviare i processi terminati.

Nell'istallazione di LHCb, il *server* del *Process Controller* è in esecuzione su un numero ridotto di PC della *farm*, denominati *control-PC*, ciascuno dei quali sovrintende a circa 200 nodi, figura (2.17).

Il *Process Controller* mantiene, per ogni nodo controllato, la lista dei processi che si vogliono in esecuzione (che chiameremo, brevemente, **processi attivi**). Esso avvia tali processi in fase di *start-up* e può riavviarli se essi terminano inaspettatamente.

Il *server* del *Process Controller* opera simultaneamente come *client* DIM e come *server* DIM.

Come *client*, esso contatta il servizio *list* e i comandi *start* e *stop* del *server* del *Task Manager* in esecuzione su ogni nodo controllato.

Come *server*, esso “pubblica” i comandi DIM “*add*” (per aggiungere un processo alla lista dei processi attivi su di uno o più nodi e avviare il processo stesso su tali nodi) e il comando DIM “*rm*” (per rimuovere un processo dalla lista dei processi attivi su di uno o più nodi e arrestare tale processo su tali nodi). Il *Process Controller* “pubblica” inoltre i servizi DIM “*ls*” (per ottenere l'elenco dei processi attivi), “*ll*” (per ottenere l'elenco dei processi attivi con i relativi parametri di avvio, ovvero *path*, *UTGID*, variabili di ambiente, *scheduler*, priorità, ecc.) e “*lss*” (per ottenere un elenco dei processi attivi che includa anche, per ogni processo, il numero di riavvii e la data e l'ora di ogni riavvio).

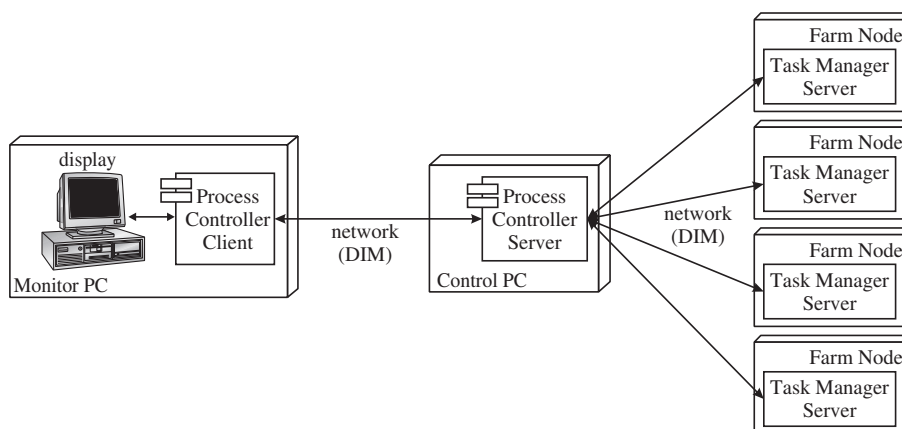


Figura 2.17: Schema dell'installazione del *Process Controller*.

2.8.1 Il riavvio rapido dei processi

Il *server* del *Process Controller*, in collaborazione con i *server* del *Task Manager*, fornisce un meccanismo per il riavvio rapido dei processi terminati inaspettatamente, figura (2.18), che qui descriviamo schematicamente.

1. Se un processo attivo su di un determinato nodo termina inaspettatamente, il *kernel* del sistema operativo di tale nodo invia un segnale SIGCHLD al *server* del *Task Manager* in esecuzione su tale nodo (genitore del processo terminato), il quale è normalmente in stato di attesa di segnali (*system call sigtimedwait* [31]).
2. Ricevuto il segnale SIGCHLD, il *server* del *Task Manager* esce immediatamente dallo stato di attesa e aggiorna il servizio DIM “list”, contenente la lista dei processi in esecuzione.
3. Il *server* del *Process Controller* — che ha sottoscritto il servizio “list” presso il *server* del *Task Manager* di tale nodo — in seguito all’aggiornamento del servizio (nella procedura di *call-back*) confronta la lista dei processi in esecuzione su tale nodo con la lista dei processi attivi sul medesimo nodo, riscontrando, in questo modo, che un processo attivo non è più in esecuzione.
4. Innescato da tale riscontro, il *server* del *Process Controller* invia il comando **start** al *Task Manager* in esecuzione sul nodo per riavviare il processo terminato.

In questo modo il meccanismo di riavvio dei processi terminati è innescato dal segnale SIGCHLD e procede direttamente, senza dover attendere i tempi di alcun tipo di controllo periodico (*polling*).

2.8.2 Il controllo del riavvio

Il *server Process Controller* è dotato di un meccanismo che impedisce il continuo riavvio di processi che terminano immediatamente a causa di qualche circostanza che deve essere corretta.

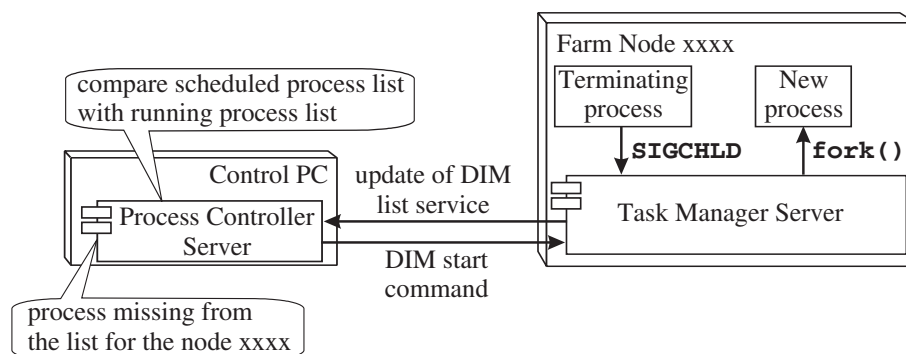


Figura 2.18: Diagramma del meccanismo di riavvio rapido dei processi.

Il meccanismo fa sì che, se un processo è riavviato più di N_{max} volte in un intervallo di tempo di T_C secondi, allora la procedura di riavvio automatico è sospesa per T_D secondi.

I valori predefiniti (valori diversi possono essere assegnati come argomenti del comando `add`) sono: $N_{max} = 10$, $T_C = 10$ s e $T_D = 300$ s.

Se N_{max} è impostato al valore -1 , allora il controllo del riavvio è soppresso e il processo può essere riavviato indefinitamente.

Se invece T_D è impostato al valore -1 , allora la procedura di riavvio automatico del processo, una volta sospesa, non è più riabilitata.

2.9 Il sistema di Monitor

Scopo del sistema di monitor è tenere sotto controllo tutti gli indicatori rilevanti per una corretta operatività della *farm*.

Il sistema di monitor, con una periodicità di poche decine di secondi (eventualmente modificabile da linea di comando all'avvio), recupera tali parametri dal *kernel* del sistema operativo di ogni nodo della *farm* e li raccoglie centralmente per poterli mostrare in tabelle o grafici sullo schermo di un PC separato (*monitor PC*) e per potere innescare gli appropriati allarmi nella FSM dell'esperimento nel caso in cui si presentino condizioni di errore o semplicemente condizioni critiche [65].

I *server* del sistema di monitor — che necessariamente devono essere in esecuzione su ogni nodo della *farm* — sono stati perciò implementati come processi “leggeri”, scritti (così come i *server* precedentemente descritti delle altre componenti del FMC) nel linguaggio ISO C99, ponendo molta attenzione al risparmio delle risorse per non sovraccaricare i nodi, già pesantemente sfruttati dagli algoritmi di selezione del *trigger*.

Si è valutato che la scrittura del codice in un linguaggio di basso livello, come il C, piuttosto che in un linguaggio di alto livello, come Java o il C++, potesse rappresentare una buona disciplina per risparmiare risorse, in quanto l'allocazione delle risorse è meno trasparente (deve essere eseguita “a mano”). Per questo motivo il codice dei *server* di FMC è stato scritto in C (mentre, per esempio, il codice dei *client* a linea di comando è scritto in C++).

Sebbene possa trattarsi di una questione di punti di vista, vorrei motivare con due esempi la scelta degli sviluppatori.

- **Allocazione della memoria.** Si vuole minimizzare non soltanto l'uso di memoria, ma anche il numero di chiamate di sistema per l'allocazione dinamica della memoria (`brk` [66] in Linux) le quali consumano CPU. Ove ragionevole, si preferisce perciò non rilasciare la memoria allocata, limitandosi a riallocarla se la domanda di memoria aumenta. Al più si può rilasciare la memoria allocata se essa non è utilizzata per un lungo periodo. In questo modo la quantità di memoria allocata converge verso la massima memoria richiesta in condizioni di funzionamento tipiche del sistema (che comunque è molto inferiore alla massima memoria richiesta in linea di principio).
- **Accesso al *proc file system*.** Ogni volta che un *i-node* del *proc file system* è acceduto — sia per la lettura di un solo carattere, sia per la lettura di un grande blocco di dati — una procedura di *handling* è invocata dal *kernel* del sistema operativo per radunare al volo l'intero insieme di dati puntati dall'*i-node*. D'altro canto, le *library iostream* (*file I/O* di alto livello in C++) e `stdio` (*file I/O* di alto livello in C) leggono i dati da un *file* aperto fino a riempire un *buffer* allocato in maniera trasparente, quindi accedono nuovamente al *file* per leggere i dati successivi, ripetendo l'operazione tante volte quante sono necessarie per leggere il *file* intero, rifornendo ogni volta lo stesso *buffer*. Se si utilizzano queste *library* per accedere agli *i-node* del *proc file system* e i dati che debbono essere letti eccedono la dimensione del *buffer* trasparentemente allocato, allora la procedura di *handling* è chiamata più di una volta dal *kernel* (tante volte quanti sono gli accessi di lettura all'*i-node*). Questo richiede più lavoro di CPU e può produrre insiemi incoerenti di dati. Per accedere ai dati del *proc file system* in una sola lettura — come desiderabile — occorre invece utilizzare le chiamate di *file I/O* di basso livello (`open` [50] e `read` [34]) con il *buffer* di lettura dimensionato in modo da poter contenere l'intero *i-node*.

I *server* del sistema di monitor chiamano periodicamente un *sensore*, cioè una procedura che recupera le informazioni dal sistema operativo, e aggiornano i dati pubblicati, raggruppati in uno o più servizi DIM.

Molto spesso i dati messi a disposizione del sistema operativo sono contatori di numeri interi (per esempio il numero di *frame* Ethernet, il numero di *interrupt*, il tempo di CPU usato per diversi tipi di compiti espresso in `USER_HZ`, ecc.). Per produrre valori significativi da questi contatori, i sensori elaborano dati sotto forma di ratei (per esempio il numero di *frame* Ethernet ricevuti per unità di tempo) e frazioni di errore (per esempio il rapporto tra il numero di *frame* Ethernet persi dall'interfaccia di rete a causa del sovraccarico della FIFO e il numero totale dei *frame* Ethernet ricevuti).

Assieme ai valori istantanei, che sono di fatto i valori medi nell'intervallo di tempo che intercorre fra due successive chiamate di aggiornamento del servizio, molti sensori pubblicano anche valori medi e valori massimi. La valutazione dei valori medi e massimi è eseguita a partire dall'avvio del processo di monitor ma può essere azzerata per mezzo di un comando DIM. Questo può essere utile, per esempio, per vedere se un certo tipo di errore è stato eliminato dopo che è stato preso un provvedimento per risolvere il problema.

2.9.1 I file system /proc e /sys

Attualmente i sensori del sistema di monitor recuperano i dati dal sistema operativo accedendo agli *i-node* del */proc file system (procfs)* [42]. Fino al *kernel* 2.4, il *procfs* era l'unica interfaccia di tipo *file system* alle strutture dati interne del *kernel*.

A partire dal *kernel* 2.5 è stata aggiunta l'interfaccia *sysfs* [67][68], allo scopo di raccogliere tutti i dati che riguardano i dispositivi (reali o virtuali) e la loro interconnettività all'interno del sistema operativo.

Molti dati stanno per essere spostati dal *procfs* al *sysfs*. La tendenza, per le future versioni del *kernel*, è di lasciare nel *procfs* soltanto le informazioni specifiche dei processi.

Attualmente, nel *kernel* 2.6, le informazioni sui contatori dell'interfaccia di rete sono accessibili in entrambi *procfs* e *sysfs*, mentre i contatori dello stack TCP/IP, dello stato della CPU e dell'uso della memoria sono accessibili soltanto nel *procfs*.

Probabilmente, in futuro, la maggior parte dei sensori del sistema di monitor dovranno essere modificati per accedere al *sysfs*.

2.9.2 I parametri soggetti al monitor

I parametri soggetti al monitor dell'FMC includono:

- Il tipo e le caratteristiche delle CPU: marca, modello e sottoversione, frequenza di *clock*, dimensioni della *cache*, prestazioni (misurate in *bogomips* [70]).
- Le caratteristiche del sistema operativo: tipo (Linux/Windows), distribuzione (Scientific Linux CERN SLC release 4.6, Scientific Linux SL release 4.4, ecc.), versione del *kernel* (2.6.9-67.0.4.EL.cernsmp, 2.6.9-42.0.3.ELsmp, ecc.), architettura (i386/x86_64).
- Statistiche sull'attività delle CPU: ratei di *context switch*, percentuale di tempo trascorsa in modalità *user mode/kernel mode*, percentuale di tempo trascorsa a servire *interrupt hard* e *soft* e in attesa per operazioni di I/O.
- Statistiche sulla memoria: memoria e memoria bassa usata/libera, gestione della memoria virtuale, *paging*, *memory overbooking*, *disk cache*.
- Statistiche sugli *interrupt hardware*.
- Statistiche sulle interfacce di rete: ratei di I/O in Byte/s e in *frame/s*, numero medio di Byte per *frame*, frazioni di errore.
- Statistiche sulla coalescenza delle interfacce di rete (rapporto tra il numero di *frame* inviati e ricevuti e il numero di *interrupt* prodotti dalle interfacce di rete).
- Statistiche sul funzionamento dello stack TCP/IP: ratei di I/O in datagrammi/s e segmenti/s, frammentazione e riassemblaggio dei datagrammi, frazioni di errore.

- Statistiche sui processi in esecuzione: TGID, UTGID, uso della CPU e della memoria, tipo di *scheduler*, priorità, numero di *thread*, CPU sulla quale il processo è attualmente in esecuzione, ecc.
- Quantità di spazio disco disponibile sui *file system* locali sia per utenti privilegiati sia per utenti non privilegiati.
- Statistiche riguardanti lo stato di integrità del disco ottenute tramite la tecnologia SMART (Self Monitorig Analysis and Reporting Technology) [71] [72].

2.10 Conclusioni

Il sistema di Monitor e di Controllo della Event Filter Farm, integralmente progettato e realizzato dal gruppo LHCb del Dipartimento di Fisica dell'Alma Mater Studiorum – Università di Bologna e della Sezione di Bologna dell'INFN, è un insieme di applicazioni particolarmente adatto al monitor e al controllo di grandi *farm on-line* di PC che eseguono applicazioni *data-intensive* in modalità *soft real-time*.

Il *software* è stato progettato per la *Event Filter Farm* dell'esperimento LHCb — composta di circa 2000 PC che processano un flusso aggregato di dati pari a circa 50 GiB/s — ma è stato in seguito incluso nel progetto JCOP del CERN e adottato da altre collaborazioni.

Capitolo 3

Il decadimento $B_s^0 \rightarrow J/\psi \phi$

3.1 Introduzione

Il decadimento $B_s^0 \rightarrow J/\psi \phi$, è giudicato essere particolarmente importante per evidenziare eventuali effetti di nuova fisica, oltre il Modello Standard, che potrebbero emergere a causa della oscillazione $B_s^0 \leftrightarrow \bar{B}_s^0$. Da un lato sulla base del Modello Standard è possibile prevedere con molta precisione il valore delle ampiezze che intervengono in questo decadimento, dall'altro vi sono teorie che estendono il Modello Standard, le quali prevedono l'esistenza di effetti che altererebbero le previsioni teoriche ordinarie. Dal punto di vista sperimentale la rivelazione dello stato finale costituito dalle due particelle J/ψ e ϕ è relativamente agevole. Una complicazione nell'attuazione della misura degli osservabili fisici collegati alla violazione della simmetria CP , deriva dal fatto che lo stato finale non ha un valore di CP definito, ma è piuttosto una miscela di autostati di CP differente ($CP = +1, -1$) che debbono essere distinti perché nella sovrapposizione degli stati non siano diluiti gli effetti di violazione della simmetria che si desiderano misurare.

Per distinguere su base statistica i due possibili stati di CP occorre realizzare un'analisi che tenga conto della distribuzione angolare dei prodotti di decadimento del processo $B_s^0 \rightarrow J/\psi (\mu^+ \mu^-) \phi (K^+ K^-)$. In questo capitolo descriveremo la procedura in corso di elaborazione, per affrontare il problema della determinazione della fase di miscelamento dei mesoni B_s in questo canale nella collaborazione LHCb.

3.2 Evoluzione temporale dei mesoni B

In questo paragrafo ci proponiamo di studiare l'evoluzione temporale di un mesone B neutro di tipo B_d o B_s , tenendo conto del fatto che i mesoni neutri, indicati nel seguito generalmente come B_q , possono oscillare e decadere. Il decadimento avviene tramite processi elettrodeboli $\Delta B = 1$, mentre l'oscillazione, che è il fenomeno di trasformazione particella-antiparticella, avviene mediante una transizione con variazione di flavour $\Delta B = 2$.

Definiamo lo stato del mesone B neutro al generico istante di tempo t mediante

il vettore dello spazio di Hilbert $|\psi(t)\rangle$:

$$|\psi(t)\rangle = \alpha(t) |B^0\rangle + \beta(t) |\bar{B}^0\rangle + \sum_f c_f(t) |f\rangle \quad (3.1)$$

con questa espressione si intende dire che, per effetto della dinamica del sapore dei quark costituenti il mesone, al generico istante di tempo t , lo stato di particella sarà una sovrapposizione degli stati di particella coniugati di carica B_q e \bar{B}_q , e degli stati $|f\rangle$, nei quali è possibile avvenga il decadimento. La base degli stati è completa e per effetto dell'evoluzione si ha ad ogni istante:

$$|\alpha(t)|^2 + |\beta(t)|^2 + \sum_f |c_f(t)|^2 = 1 \quad (3.2)$$

Nel sistema di riferimento in cui la particella è a riposo lo stato (3.1) evolve secondo l'equazione non relativistica di Schrödinger:

$$i\hbar \frac{\partial}{\partial t} |\psi\rangle = \mathcal{H} |\psi(t)\rangle \quad (3.3)$$

dove \mathcal{H} è l'operatore hermitiano che agisce nello spazio di Hilbert degli stati e descrive l'effetto delle interazioni fondamentali sui costituenti della particella:

$$\mathcal{H} = \mathcal{H}_{weak} + \mathcal{H}_{strong} + \mathcal{H}_{em} \quad (3.4)$$

con

$$\mathcal{H}_{weak} \ll \mathcal{H}_0 = \mathcal{H}_{strong} + \mathcal{H}_{em} \quad (3.5)$$

Con quest'ultima relazione d'ordine intendiamo dire che le interazioni deboli sono molto meno intense di quelle forti e di quelle elettromagnetiche. Per risolvere il problema dell'evoluzione temporale (3.3) si ricorre all'approssimazione di Wigner-Weisskopf [73], secondo la quale gli stati $|f\rangle$ di decadimento sono stabili rispetto all'interazione \mathcal{H}_{weak} (si ricorre in sostanza ad uno sviluppo perturbativo del secondo ordine in \mathcal{H}_{weak}). In questo modo si ottiene una equazione di evoluzione per lo stato di particella nella base dei due stati $|B^0\rangle$ e $|\bar{B}^0\rangle$. Posto:

$$|\psi(t)\rangle = a(t) |B^0\rangle + b(t) |\bar{B}^0\rangle \quad (3.6)$$

l'equazione di evoluzione diventa:

$$i\hbar \frac{\partial}{\partial t} |\psi\rangle = \mathbf{H} |\psi(t)\rangle \quad (3.7)$$

con

$$|a(t)|^2 + |b(t)|^2 \neq 1 \quad (3.8)$$

Osserviamo che l'operatore di evoluzione \mathcal{H} (eq. 3.6) può essere scritto come matrice 2×2 :

$$\mathbf{H} = \mathbf{M} - \frac{i}{2} \mathbf{\Gamma} = \begin{pmatrix} m_{11} - \frac{i}{2} \Gamma_{11} & m_{12} - \frac{i}{2} \Gamma_{12} \\ m_{21} - \frac{i}{2} \Gamma_{21} & m_{22} - \frac{i}{2} \Gamma_{22} \end{pmatrix} \quad (3.9)$$

con

$$\mathbf{M} = \mathbf{M}^\dagger \quad (3.10)$$

e

$$\mathbf{\Gamma} = \mathbf{\Gamma}^\dagger \quad (3.11)$$

L'operatore \mathbf{H} non è hermitiano, mentre lo sono le sue due componenti \mathbf{M} e $\mathbf{\Gamma}$, i cui autovalori saranno dunque reali. I valori di \mathbf{M} sono:

$$m_{11} = m_0 + \langle B | H_{weak} | B \rangle + \mathbf{P} \sum_f \frac{\langle B | H_{weak} | f \rangle \langle f | H_{weak} | B \rangle}{m_0 - E_f} \quad (3.12)$$

e

$$m_{22} = m_0 + \langle \bar{B} | H_{weak} | \bar{B} \rangle + \mathbf{P} \sum_f \frac{\langle \bar{B} | H_{weak} | f \rangle \langle f | H_{weak} | \bar{B} \rangle}{m_0 - E_f} \quad (3.13)$$

con

$$m_{12} = \langle B | H_{weak} | \bar{B} \rangle + \mathbf{P} \sum_f \frac{\langle B | H_{weak} | f \rangle \langle f | H_{weak} | \bar{B} \rangle}{m_0 - E_f} \quad (3.14)$$

gli stati $|f\rangle$ sono tutti gli stati accessibili a B_q e \bar{B}_q reali e virtuali. I valori di $\mathbf{\Gamma}$ sono invece:

$$\Gamma_{11} = 2\pi \sum_f |\langle B | H_{weak} | f \rangle|^2 \delta(m_0 - E_f) \quad (3.15)$$

e

$$\Gamma_{22} = 2\pi \sum_f |\langle \bar{B} | H_{weak} | f \rangle|^2 \delta(m_0 - E_f) \quad (3.16)$$

con

$$\Gamma_{12} = 2\pi \sum_f \langle B | H_{weak} | f \rangle \langle f | H_{weak} | \bar{B} \rangle \delta(m_0 - E_f) \quad (3.17)$$

dove gli stati $|f\rangle$ sono solo quelli reali, accessibili per decadimento. Abbiamo che:

$$m_{21} = m_{12}^*, \quad \Gamma_{21} = \Gamma_{12}^* \quad (3.18)$$

Inoltre, si può dimostrare che per ragioni fisiche fondamentali (validità del teorema CPT) si può assumere

$$m = m_{11} = m_{22}, \quad \Gamma = \Gamma_{11} = \Gamma_{22} \quad (3.19)$$

per cui possiamo scrivere l'espressione per \mathbf{H} (3.9) nella forma seguente:

$$\mathbf{H} = \begin{pmatrix} m - \frac{i}{2}\Gamma & m_{12} - \frac{i}{2}\Gamma_{12} \\ m_{12}^* - \frac{i}{2}\Gamma_{12}^* & m - \frac{i}{2}\Gamma \end{pmatrix} \quad (3.20)$$

Passiamo ora a risolvere il problema agli autovalori per (3.20). Si deve avere:

$$\begin{vmatrix} m - \frac{i}{2}\Gamma - \lambda & m_{12} - \frac{i}{2}\Gamma_{12} \\ m_{12}^* - \frac{i}{2}\Gamma_{12}^* & m - \frac{i}{2}\Gamma - \lambda \end{vmatrix} = 0 \quad (3.21)$$

da cui otteniamo:

$$\lambda^2 - 2\lambda H - (H_{21}H_{12} - H) = 0 \quad (3.22)$$

dove si è posto per semplicità

$$H = m - \frac{i}{2}\Gamma, \quad H_{12} = m_{12} - \frac{i}{2}\Gamma_{12} \quad (3.23)$$

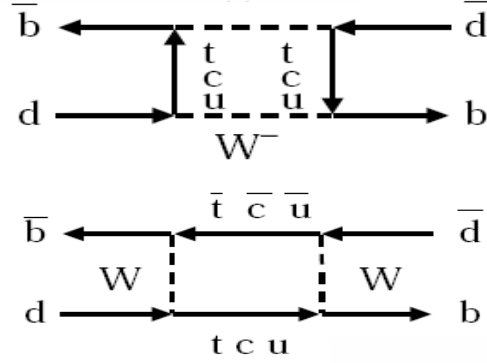


Figura 3.1: Diagrammi di Feynman relativi all'oscillazione $B \leftrightarrow \bar{B}$.

Le due soluzioni del problema agli autovalori sono:

$$\lambda_{\pm} = H \pm \sqrt{H_{12}H_{21}} \equiv H \pm H_{12} \frac{q}{p} \quad (3.24)$$

Abbiamo introdotto il numero complesso:

$$\frac{q}{p} = e^{in\pi} \sqrt{\frac{H_{21}}{H_{12}}} = e^{in\pi} \sqrt{\frac{m_{12}^* - \frac{i}{2}\Gamma_{12}^*}{m_{12} - \frac{i}{2}\Gamma_{12}}} \quad (3.25)$$

con $n = 0, 1$. I due autostati di propagazione libera (stiamo studiando l'evoluzione in assenza di interazione esterna) corrispondenti agli autovalori (3.24) sono:

$$|B_+\rangle = \frac{|B^0\rangle + \frac{q}{p} |\bar{B}^0\rangle}{\sqrt{1 + \left(\frac{|q|}{|p|}\right)^2}} \quad (3.26)$$

$$|B_-\rangle = \frac{|B^0\rangle - \frac{q}{p} |\bar{B}^0\rangle}{\sqrt{1 + \left(\frac{|q|}{|p|}\right)^2}} \quad (3.27)$$

Una soluzione generale del problema si può dunque scrivere nella forma seguente:

$$|\psi(t)\rangle = ae^{i\lambda_+t} |B_+\rangle + be^{i\lambda_-t} |B_-\rangle \quad (3.28)$$

Siamo interessati a determinare l'evoluzione di stati di particella preparati all'istante iniziale in uno stato di sapore determinato per cui imponiamo le condizioni iniziali seguenti:

$$|B(t)\rangle \equiv |\psi(t)\rangle \quad \text{quando} \quad |\psi(0)\rangle = |B^0\rangle \quad (3.29)$$

e

$$|\bar{B}(t)\rangle \equiv |\psi(t)\rangle \quad \text{quando} \quad |\psi(0)\rangle = |\bar{B}^0\rangle \quad (3.30)$$

Nei due casi si ottengono rispettivamente le espressioni dell'equazione di evoluzione temporale dello stato fisico di particella e di antiparticella.

$$|B_q(t)\rangle = \frac{1}{2} (e^{i\lambda_+t} + e^{i\lambda_-t}) |B^0\rangle + \frac{1}{2} (e^{i\lambda_+t} - e^{i\lambda_-t}) \frac{q}{p} |\bar{B}^0\rangle \quad (3.31)$$

In analogia, per l'evoluzione temporale per lo stato fisico che all'istante $t = 0$ è nello stato di antiparticella si ha:

$$|\bar{B}_q(t)\rangle = \frac{1}{2} (e^{i\lambda_+t} - e^{i\lambda_-t}) \frac{p}{q} |B^0\rangle + \frac{1}{2} (e^{i\lambda_+t} + e^{i\lambda_-t}) |\bar{B}^0\rangle \quad (3.32)$$

Introduciamo per comodità le funzioni:

$$g_{\pm} = \frac{1}{2} (e^{i\lambda_+t} \pm e^{i\lambda_-t}) \quad (3.33)$$

e riscriviamo le espressioni (3.31) e (3.32) sinteticamente:

$$|B_q(t)\rangle = g_+(t) |B^0\rangle + \frac{q}{p} g_-(t) |\bar{B}^0\rangle \quad (3.34)$$

$$|\bar{B}_q(t)\rangle = \frac{p}{q} g_-(t) |B^0\rangle + g_+(t) |\bar{B}^0\rangle \quad (3.35)$$

I valori della massa e della larghezza di decadimento (inverso della vita media dello stato) si possono definire mediante l'equazione seguente:

$$\lambda_{\pm} \equiv M_{\pm} - \frac{i}{2} \Gamma_{\pm} = H_0 \pm H_{12} \frac{p}{q} \quad (3.36)$$

dove:

$$M_{\pm} \equiv \Re(\lambda_{\pm}) = M \pm \Re\left(H_{12} \frac{p}{q}\right) \quad (3.37)$$

$$\Gamma_{\pm} \equiv -2\Im(\lambda_{\pm}) = \Gamma \mp \Im\left(H_{12} \frac{p}{q}\right) \quad (3.38)$$

Per avere una lettura più chiara del formalismo riscriviamo il valore delle funzioni $g_{\pm}(t)$. Definendo

$$\lambda = \lambda_+ + \lambda_- \quad , \quad \Delta\lambda = \lambda_+ - \lambda_- \quad \Delta M = M_+ - M_- \quad (3.39)$$

si ottiene:

$$\lambda = M_+ + M_- - \frac{i}{2} (\Gamma_+ + \Gamma_-) = 2 \left(M - \frac{i}{2} \Gamma \right) \quad (3.40)$$

$$\Delta\lambda = \Delta M - \frac{i}{2} \Delta\Gamma \quad (3.41)$$

in modo da poter scrivere l'espressione (3.33) come:

$$g_+(t) = e^{iMt} e^{-\frac{1}{2}\Gamma t} \left[\cos\left(\frac{\Delta M}{2}t\right) \cosh\left(\frac{\Delta\Gamma}{4}t\right) - i \sin\left(\frac{\Delta M}{2}t\right) \sinh\left(\frac{\Delta\Gamma}{4}t\right) \right] \quad (3.42)$$

e

$$g_-(t) = e^{iMt} e^{-\frac{1}{2}\Gamma t} \left[-\cos\left(\frac{\Delta M}{2}t\right) \sinh\left(\frac{\Delta\Gamma}{4}t\right) + i \sin\left(\frac{\Delta M}{2}t\right) \cosh\left(\frac{\Delta\Gamma}{4}t\right) \right] \quad (3.43)$$

Per gli sviluppi successivi è importante chiarire il ruolo del rapporto $\frac{q}{p}$. Nell'ambito del Modello Standard si può stimare il valore del rapporto fra Γ_{12} e m_{12} che risulta essere:

$$\left| \frac{\Gamma_{12}}{m_{12}} \right| \propto \frac{m_b^2}{m_t^2} \ll 1 \quad (3.44)$$

per cui:

$$\begin{aligned} \frac{q}{p} &= \sqrt{\frac{m_{12}^*}{m_{12}}} \sqrt{\frac{1 - \frac{i}{2} \frac{\Gamma_{12}^*}{m_{12}^*}}{1 - \frac{i}{2} \frac{\Gamma_{12}}{m_{12}}}} \\ &= e^{-i(\arg(m_{12}) + \pi)} \cdot \left(1 - \frac{1}{2} \Im \left(\frac{\Gamma_{12}}{m_{12}} \right) \right) \end{aligned} \quad (3.45)$$

È evidente dunque che con ottima approssimazione si può ritenere si abbia:

$$\left| \frac{q}{p} \right| = 1 \quad (3.46)$$

mentre la fase è sostanzialmente legata all'elemento di matrice m_{12} , e nella rappresentazione di Wolfenstein della matrice CKM otteniamo:

$$\arg(m_{12}) = \arg(V_{tb}V_{tq}^*)^2 = -2 \arg(V_{tq}^*) \quad q = d, s \quad (3.47)$$

, per cui si ha rispettivamente per B_d e B_s :

$$\frac{q}{p} = e^{-i(\pi + 2 \arg(V_{td}))} \quad (3.48)$$

$$\frac{q}{p} = e^{-i(\pi + 2 \arg(V_{ts}))} \quad (3.49)$$

Notiamo in conclusione di paragrafo che il valore del rapporto $\frac{q}{p}$ è legato direttamente alla violazione della simmetria CP . Vedremo in dettaglio questo punto nel paragrafo seguente. Osserviamo ora che se si avesse $\frac{q}{p} = 1$ allora i due autostati di propagazione libera sarebbero anche autostati di CP essendo costituiti in eguale misura dai due stati coniugati B_q e \bar{B}_q come mostrano le equazioni per gli autostati (3.26) e (3.27). Immediata conseguenza sarebbe dunque che $[H, CP] = 0$ cioè si avrebbe conservazione di CP . Ecco dunque perché nella massima generalità il valore del rapporto $\frac{q}{p} \neq 1$.

3.3 Decadimento istantaneo dei mesoni B

Consideriamo ora il caso del decadimento dei mesoni neutri B_q^0 e \bar{B}_q^0 nello stesso stato finale f . Introduciamo le ampiezze di transizione per gli stati $|B_q^0\rangle$ e $|\bar{B}_q^0\rangle$

$$A_f \equiv \langle f | \mathbf{H}_{\text{eff}} | B_q^0 \rangle, \quad \bar{A}_f \equiv \langle f | \mathbf{H}_{\text{eff}} | \bar{B}_q^0 \rangle \quad (3.50)$$

Dove H_{eff} è l'operatore hamiltoniano effettivo $|\Delta B| = 1$ responsabile del decadimento [74]. Utilizzando le relazioni (3.34) e (3.35) possiamo scrivere le espressioni delle ampiezze di decadimento istantaneo:

$$A(B_q(t) \rightarrow f) = g_+(t)A_f + \frac{q}{p}g_-(t)\bar{A}_f = A_f(g_+(t) + \lambda_f g_-(t)) \quad (3.51)$$

$$A(\bar{B}_q(t) \rightarrow f) = \frac{p}{q} g_-(t) A_f + g_+(t) \bar{A}_f = A_f \frac{p}{q} (g_-(t) + \lambda_f g_+(t)) \quad (3.52)$$

dove abbiamo posto:

$$\lambda_f \equiv \frac{q \bar{A}_f}{p A_f} \quad (3.53)$$

Per lo stato CP coniugato di $|f\rangle$ che indichiamo con $|\bar{f}\rangle$ le espressioni delle ampiezze diventano:

$$A(\bar{B}_q(t) \rightarrow \bar{f}) = \bar{A}_{\bar{f}} \left(g_+(t) + \frac{1}{\lambda_{\bar{f}}} g_-(t) \right) \quad (3.54)$$

$$A(B_q(t) \rightarrow \bar{f}) = \bar{A}_{\bar{f}} \left(\frac{1}{\lambda_{\bar{f}}} g_+(t) + g_-(t) \right) \quad (3.55)$$

dove abbiamo posto:

$$\lambda_{\bar{f}} \equiv \frac{q \bar{A}_{\bar{f}}}{p A_{\bar{f}}} \quad (3.56)$$

Per calcolare i rate di decadimento istantanei dobbiamo calcolare prima il valore del modulo quadrato della ampiezze. Abbiamo allora:

$$|A(B_q(t) \rightarrow f)|^2 = |A_f|^2 \left(|g_+(t)|^2 + |\lambda_f|^2 |g_-(t)|^2 + 2\Re(\lambda_f g_+^*(t) g_-(t)) \right) \quad (3.57)$$

Analogamente per l'altra ampiezze quadrata otteniamo:

$$|A(\bar{B}_q(t) \rightarrow f)|^2 = |A_{\bar{f}}|^2 \left| \frac{p}{q} \right|^2 \left(|g_-(t)|^2 + |\lambda_{\bar{f}}|^2 |g_+(t)|^2 + 2\Re(\lambda_{\bar{f}} g_-^*(t) g_+(t)) \right) \quad (3.58)$$

Per gli stati che decadono nello stato $|\bar{f}\rangle$ abbiamo:

$$|A(\bar{B}_q(t) \rightarrow \bar{f})|^2 = |\bar{A}_{\bar{f}}|^2 \left(|g_+(t)|^2 + |\lambda_{\bar{f}}|^{-2} |g_-(t)|^2 + 2\Re(\lambda_{\bar{f}}^{-1} g_+(t) g_-^*(t)) \right) \quad (3.59)$$

$$|A(B_q(t) \rightarrow f)|^2 = |\bar{A}_{\bar{f}}|^2 \left| \frac{q}{p} \right|^2 \left(|\lambda_{\bar{f}}|^{-2} |g_+(t)|^2 + |g_-(t)|^2 + 2\Re(\lambda_{\bar{f}}^{-1} g_+(t) g_-^*(t)) \right) \quad (3.60)$$

I rate istantanei di decadimento si possono esprimere in funzione dei moduli quadrati delle ampiezze di decadimento secondo relazioni del tipo:

$$\Gamma(B_q \rightarrow f) \equiv N_f \cdot |A(B_q(t) \rightarrow f)|^2 \quad (3.61)$$

dove la costante N_f deve essere determinata, se necessario, dalle condizioni cinematiche relative al decadimento specifico.

Possiamo riscrivere le relazioni per i rate di decadimento tenendo conto delle espressioni calcolate in precedenza per le funzioni $g_{\pm}(t)$ in funzione dei parametri fisici M , Γ , ΔM e $\Delta\Gamma$. Abbiamo calcolato il valore di $|g_{\pm}(t)|^2$, per completezza calcoliamo ora:

$$g_+^*(t) \cdot g_-(t) = \frac{1}{2} e^{-\Gamma t} \left(-\sinh\left(\frac{\Delta\Gamma t}{2}\right) + \sin(\Delta M t) \right) \quad (3.62)$$

Dalle espressioni precedenti possiamo calcolare le espressioni per i rate di decadimento istantaneo. Otteniamo:

$$\begin{aligned} \Gamma(B_q(t) \rightarrow f) = N_f |A_f^{(q)}|^2 e^{-\Gamma t} & \left[\frac{1 + |\lambda_f^{(q)}|^2}{2} \cosh\left(\frac{\Delta\Gamma t}{2}\right) + \frac{1 - |\lambda_f^{(q)}|^2}{2} \cos\left(\frac{\Delta Mt}{2}\right) + \right. \\ & \left. - \Re(\lambda_f^{(q)}) \sinh\left(\frac{\Delta\Gamma t}{2}\right) - \Im(\lambda_f^{(q)}) \sin\left(\frac{\Delta Mt}{2}\right) \right] \end{aligned} \quad (3.63)$$

$$\begin{aligned} \Gamma(\bar{B}_q(t) \rightarrow f) = N_f \left|\frac{p}{q}\right|^2 |\bar{A}_f^{(q)}|^2 e^{-\Gamma t} & \left[\frac{1 + |\lambda_f^{(q)}|^2}{2} \cosh\left(\frac{\Delta\Gamma t}{2}\right) - \frac{1 - |\lambda_f^{(q)}|^2}{2} \cos\left(\frac{\Delta Mt}{2}\right) + \right. \\ & \left. - \Re(\lambda_f^{(q)}) \sinh\left(\frac{\Delta\Gamma t}{2}\right) + \Im(\lambda_f^{(q)}) \sin\left(\frac{\Delta Mt}{2}\right) \right] \end{aligned} \quad (3.64)$$

$$\begin{aligned} \Gamma(B_q(t) \rightarrow \bar{f}) = N_f \left|\frac{q}{p}\right|^2 |\bar{A}_{\bar{f}}^{(q)}|^2 e^{-\Gamma t} & \left[\frac{1 + |\lambda_{\bar{f}}^{(q)}|^{-2}}{2} \cosh\left(\frac{\Delta\Gamma t}{2}\right) - \frac{1 - |\lambda_{\bar{f}}^{(q)}|^{-2}}{2} \cos\left(\frac{\Delta Mt}{2}\right) + \right. \\ & \left. - \Re\left(\frac{1}{\lambda_{\bar{f}}^{(q)}}\right) \sinh\left(\frac{\Delta\Gamma t}{2}\right) + \Im\left(\frac{1}{\lambda_{\bar{f}}^{(q)}}\right) \sin\left(\frac{\Delta Mt}{2}\right) \right] \end{aligned} \quad (3.65)$$

$$\begin{aligned} \Gamma(\bar{B}_q(t) \rightarrow \bar{f}) = N_f |\bar{A}_{\bar{f}}^{(q)}|^2 e^{-\Gamma t} & \left[\frac{1 + |\lambda_{\bar{f}}^{(q)}|^{-2}}{2} \cosh\left(\frac{\Delta\Gamma t}{2}\right) + \frac{1 - |\lambda_{\bar{f}}^{(q)}|^{-2}}{2} \cos\left(\frac{\Delta Mt}{2}\right) + \right. \\ & \left. - \Re\left(\frac{1}{\lambda_{\bar{f}}^{(q)}}\right) \sinh\left(\frac{\Delta\Gamma t}{2}\right) - \Im\left(\frac{1}{\lambda_{\bar{f}}^{(q)}}\right) \sin\left(\frac{\Delta Mt}{2}\right) \right] \end{aligned} \quad (3.66)$$

3.4 Il decadimento $B_s^0 \rightarrow J/\psi \phi$

Il decadimento $B_s \rightarrow J/\psi \phi$ nel Modello Standard è dominato dal processo $\bar{b} \rightarrow c\bar{c}s$. I valori dei rate, come mostrano le espressioni della sezione precedente, dipendono dal valore assunto da λ_f^s , per cui si ha:

$$\lambda_f^s \equiv \frac{q}{p} \frac{\bar{A}_f^s}{A_f^s} = e^{-i\phi_s} \cdot \eta_f \quad (3.67)$$

dove, come già ricordato:

$$\phi_s = 2 \arg(V_{ts}) \quad (3.68)$$

mentre:

$$\frac{\bar{A}_f^s}{A_f^s} = \eta_f \quad (3.69)$$

In quest'ultima espressione $\eta_f = \pm 1$ rappresenta l'autovalore di CP dello stato finale $|f\rangle$ definito dalla relazione

$$|\bar{f}\rangle = CP |f\rangle = \eta_f |f\rangle \quad (3.70)$$

Si ottengono pertanto per la parte reale e immaginaria di $\lambda_f^{(s)}$ le seguenti espressioni:

$$\Im(\lambda_f^{(s)}) = -\eta_f \sin \phi_s \quad \Re(\lambda_f^{(s)}) = \eta_f \cos \phi_s \quad (3.71)$$

Otteniamo in definitiva le seguenti ampiezze di decadimento:

$$|A_f^{(s)}(t)|^2 \propto |A_f^{(s)}|^2 e^{(-\Gamma_s t)} \left\{ \cosh \frac{\Delta\Gamma_s t}{2} - \eta_f \cos \phi_s \sinh \frac{\Delta\Gamma_s t}{2} + \eta_f \sin \phi_s \sin(\Delta M_s t) \right\} \quad (3.72)$$

$$|\bar{A}_f^{(s)}(t)|^2 \propto |A_f^{(s)}|^2 e^{(-\Gamma_s t)} \left\{ \cosh \frac{\Delta\Gamma_s t}{2} - \eta_f \cos \phi_s \sinh \frac{\Delta\Gamma_s t}{2} - \eta_f \sin \phi_s \sin(\Delta M_s t) \right\} \quad (3.73)$$

Il decadimento $B_s \rightarrow J/\psi \phi$ non ha un valore η_f di CP definito, poiché il mesone B_s di spin zero decade in due particelle di spin uno, con numeri quantici $J^{PC} = 1^{--}$. Per la conservazione del momento angolare le due particelle dello stato finale nel sistema di riferimento in cui il B_s è a riposo debbono avere un momento orbitale relativo $L = 0, 1, 2$. Gli autovalori di CP dello stato finale sono dati da:

$$CP(J/\psi)CP(J)CP(\psi)(-1)^L = +1, -1, +1. \quad (3.74)$$

Per effettuare la misura della fase di miscelamento, evitando effetti di diluizione, dovuti alla sovrapposizione di stati con autovalori di CP opposti, è necessario distinguere gli eventi in base allo stato di momento angolare relativo dei prodotti di decadimento. Si può dimostrare che l'ampiezza deve avere una espressione del tipo:

$$A(B_s \rightarrow J/\psi \phi) = A_0(M_\phi/E_\phi) \epsilon_{J/\psi}^{*L} \epsilon_\phi^{*L} - A_\parallel \epsilon_{J/\psi}^{*T} \epsilon_\phi^{*T} / \sqrt{2} - i A_\perp \epsilon_{J/\psi}^* \times \epsilon_\phi^* \cdot \hat{p}_\phi / \sqrt{2} \quad (3.75)$$

scelto come riferimento il sistema in cui la particella J/ψ è a riposo, si ha:

- $\epsilon_{J/\psi}^*$ e ϵ_ϕ^* sono i vettori di polarizzazione;
- E_ϕ è l'energia della ϕ nel sistema di riferimento della J/ψ ;
- \hat{p}_ϕ è il versore della direzione del moto della ϕ ;

Gli apici T e L si riferiscono rispettivamente alle proiezioni trasversale e longitudinale dei vettori di polarizzazione rispetto alla direzione definita da \hat{p}_ϕ . Le ampiezze dei due decadimenti corrispondenti a $CP = +1$ sono quelli che corrispondono alle ampiezze di polarizzazione A_0 e A_\parallel , mentre l'ampiezza corrispondente allo stato di $CP = -1$ è A_\perp . La realizzazione dell'analisi angolare

può essere realizzata in modo generale anche mediante il formalismo dell'elicità. Le due particelle dello stato finale J/ψ e ϕ debbono avere gli stessi stati di elicità ($\lambda = \vec{s} \cdot \vec{p} / |\vec{s} \cdot \vec{p}| = +1, 0, -1$) poiché il mesone B_s ha spin zero. Le ampiezze dell'elicità $H_-(-1, -1)$, $H_0(0, 0)$ e $H_+(+1, +1)$ normalizzate sono legate alle ampiezze di polarizzazione dalle relazioni seguenti:

$$A_0 = H_0 ; A_{\parallel} = \frac{1}{\sqrt{2}}(H_+ + H_-) ; A_{\perp} = \frac{1}{\sqrt{2}}(H_+ - H_-) \quad (3.76)$$

In assenza di fenomeni di violazione di CP per le ampiezze di decadimento al tempo $t = 0$ si ha:

$$\bar{A}_0 = A_0 ; \bar{A}_{\parallel} = A_{\parallel} ; \bar{A}_{\perp} = -A_{\perp} \quad (3.77)$$

In definitiva lo stato finale può trovarsi in tre stati di polarizzazione differenti, tali che il rateo di decadimento si può scrivere:

$$\Gamma(t) \approx |A_0(t)|^2 + |A_{\parallel}(t)|^2 + |A_{\perp}|^2 \quad (3.78)$$

3.4.1 La base trasversale

L'impiego della base trasversale facilita l'analisi angolare. Si utilizza come riferimento la direzione definita dall'impulso dei prodotti del decadimento osservato nello stato finale a quattro corpi carichi $B_s \rightarrow J/\psi(l^+l^-)\phi(K^+K^-)$. In riferimento alla figura (3.2):

- L'asse x è definito dalla direzione dell'impulso della particella ϕ nel riferimento solidale con la J/ψ ;
- L'asse z è perpendicolare al piano che contiene i prodotti del decadimento $\phi \rightarrow K^+K^-$ nel sistema solidale con la J/ψ . Il verso positivo dell'asse z è dato dal prodotto vettoriale $\hat{p}_{K^-} \wedge \hat{p}_{K^+}$, dove $\hat{p}_{K^{\pm}}$ sono i versori dei due kaoni nel sistema di riferimento solidale della J/ψ .
- L'asse y è definito nel piano individuato dai due kaoni, con $p_y(K^+) \geq 0$ nel sistema di riferimento di riferimento solidale con la J/ψ .

L'angolo ψ è dunque l'angolo relativo fra il vettore \hat{p}'_{K^+} nel sistema di riferimento della ϕ e l'asse x . Gli altri due angoli trasversali (θ, φ) descrivono la direzione del leptone positivo l^+ dalla J/ψ rispetto al sistema di riferimento della J/ψ . Il versore della direzione del leptone l^+ può essere definito cioè come:

$$\hat{n} = (n_x, n_y, n_z) = (\sin \theta \cos \varphi, \sin \theta \sin \varphi, \cos \theta) \quad (3.79)$$

Con queste convenzioni si ha dunque:

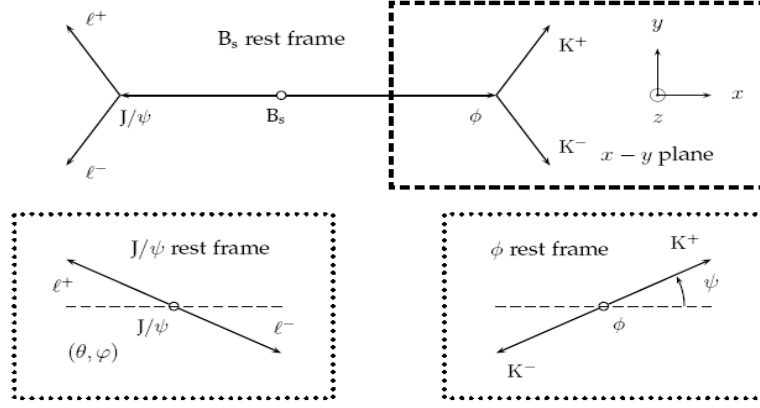
$$\begin{aligned} \hat{x} &= \hat{p}_{\phi} ; \hat{h} = \frac{\hat{p}_{K^+} - \hat{p}_{\phi}(\hat{p}_{\phi} \cdot \hat{p}_{K^+})}{|\hat{p}_{K^+} - \hat{p}_{\phi}(\hat{p}_{\phi} \cdot \hat{p}_{K^+})|} ; \hat{z} = \hat{x} \wedge \hat{y} \\ \sin \theta \cos \varphi &= \hat{p}_{l^+} \cdot \hat{x} ; \sin \theta \sin \varphi = \hat{p}_{l^+} \cdot \hat{y} ; \cos \theta = \hat{p}_{l^+} \cdot \hat{z} \end{aligned} \quad (3.80)$$

nel sistema di riferimento di J/ψ .

Si ha poi:

$$\cos \psi = -\hat{p}'_{K^+} \cdot \hat{p}'_{J/\psi} \quad (3.81)$$

dove le quantità con apice sono versori nel sistema di riferimento solidale con la particella ϕ .

Figura 3.2: Gli angoli (θ, φ, ψ) della base trasversa.

3.4.2 La distribuzione di probabilità del decadimento in un angolo

Per poter distinguere su base statistica gli eventi corrispondenti a valori di CP differenti, il requisito minimo è di definire le distribuzioni di probabilità dei decadimenti in funzione del cosiddetto angolo di trasversalità θ , definito nel paragrafo precedente.

La distribuzione angolare differenziale per il decadimento $B_s \rightarrow J/\psi \phi$ in funzione di θ è data dall'espressione seguente:

$$\frac{d\Gamma[B_s(t) \rightarrow f]}{d\cos\theta} \propto \frac{3}{8} (|A_0(t)|^2 + |A_{\parallel}(t)|^2) (1 + \cos^2\theta) + \frac{3}{4} |A_{\perp}(t)|^2 \sin^2\theta \quad (3.82)$$

Le ampiezze dipendenti dal tempo che compaiono nella espressione precedente corrispondono a quelle riportate nelle equazioni (3.72). Le ampiezze A_0 e A_{\parallel} corrispondono a $CP = +1$ cioè a $\eta_f = 1$:

$$|A_{0,\parallel}(t)|^2 \propto |A_{0,\parallel}|^2 e^{-\Gamma_s t} [Y_{0,\parallel}(t) + Y(t)] \quad (3.83)$$

l'ampiezza A_{\perp} corrisponde al caso $CP = -1$ cioè $\eta_f = -1$:

$$|A_{\perp}(t)|^2 \propto |A_{\perp}|^2 e^{-\Gamma_s t} [Y_{\perp}(t) - Y(t)] \quad (3.84)$$

dove per comodità si è posto:

$$Y_{0,\parallel}(t) = \cos \frac{\Delta\Gamma_s t}{2} - \cosh \phi_s \sin \frac{\Delta\Gamma_s t}{2} \quad (3.85)$$

$$Y_{\perp}(t) = \cos \frac{\Delta\Gamma_s t}{2} + \cosh \phi_s \sin \frac{\Delta\Gamma_s t}{2} \quad (3.86)$$

$$Y(t) = \sin \phi_s + \sin \Delta m_s t \quad (3.87)$$

Sviluppando l'espressione (3.82), grazie alle ultime relazioni introdotte, otteniamo:

$$\begin{aligned} \frac{d\Gamma [B_s(t) \rightarrow J/\psi \phi]}{d \cos \theta} &= \frac{3}{8} [|A_0|^2 + |A_{\parallel}|^2] e^{-\Gamma_s t} [Y_{0,\parallel}(t) + Y(t)] (1 + \cos^2 \theta) \\ &+ \frac{3}{4} |A_{\perp}(t)|^2 [Y_{\perp}(t) - Y(t)] (1 - \cos^2 \theta) \end{aligned} \quad (3.88)$$

Eseguiamo i medesimi calcoli in riferimento al processo $\bar{B}_s(t) \rightarrow J/\psi \phi$, tenendo conto del fatto che:

$$|\bar{A}_{0,\parallel}(t)|^2 \propto |A_{0,\parallel}|^2 e^{-\Gamma_s t} [Y_0(t) - Y(t)] \quad (3.89)$$

$$|\bar{A}_{\perp}(t)|^2 \propto |A_{\perp}|^2 e^{-\Gamma_s t} [Y_{\perp}(t) + Y(t)] \quad (3.90)$$

Abbiamo quindi:

$$\begin{aligned} \frac{d\Gamma [\bar{B}_s(t) \rightarrow J/\psi \phi]}{d \cos \theta} &= \frac{3}{8} [|A_0|^2 + |A_{\parallel}|^2] e^{-\Gamma_s t} [Y_{0,\parallel}(t) - Y(t)] (1 + \cos^2 \theta) \\ &+ \frac{3}{4} |A_{\perp}|^2 [Y_{\perp}(t) + Y(t)] (1 - \cos^2 \theta) \end{aligned} \quad (3.91)$$

Teniamo ora conto anche dell'effetto del tagging che a causa della probabilità non nulla di fornire una classificazione errata, in misura pari a ω_{tag} combina i due processi $B_s(t) \rightarrow J/\psi \phi$ e $\bar{B}_s(t) \rightarrow J/\psi \phi$, in maniera tale che i rate osservabili risultano:

$$R(t, \cos \theta) \propto (1 - \omega_{tag}) \frac{d\Gamma [B_s(t) \rightarrow J/\psi \phi]}{d \cos \theta} + \omega_{tag} \frac{d\Gamma [\bar{B}_s(t) \rightarrow J/\psi \phi]}{d \cos \theta} \quad (3.92)$$

e

$$\bar{R}(t, \cos \theta) \propto (1 - \omega_{tag}) \frac{d\Gamma [\bar{B}_s(t) \rightarrow J/\psi \phi]}{d \cos \theta} + \omega_{tag} \frac{d\Gamma [B_s(t) \rightarrow J/\psi \phi]}{d \cos \theta} \quad (3.93)$$

le equazioni 3.92 e 3.93 possono essere riscritte esplicitamente come:

$$\begin{aligned} R(t, \cos \theta) &\propto \frac{3}{8} [|A_0|^2 + |A_{\parallel}|^2] e^{-\Gamma_s t} [Y_{0,\parallel}(t) + Y(t)] (1 + \cos^2 \theta) \\ &+ \frac{3}{4} |A_{\perp}|^2 [Y_{\perp}(t) - Y(t)] (1 - \cos^2 \theta) \\ &+ \frac{3}{8} [|A_0|^2 + |A_{\parallel}|^2] e^{-\Gamma_s t} [-2\omega Y(t)] (1 + \cos^2 \theta) \\ &+ \frac{3}{4} |A_{\perp}|^2 [2\omega Y(t)] \sin^2 \theta \\ &= \frac{3}{8} [|A_0|^2 + |A_{\parallel}|^2] e^{-\Gamma_s t} [Y_{0,\parallel}(t) + (1 - 2\omega)Y(t)] (1 + \cos^2 \theta) \\ &+ \frac{3}{4} |A_{\perp}|^2 [Y_{\perp}(t) - (1 - 2\omega)Y(t)] (1 - \cos^2 \theta) \end{aligned} \quad (3.94)$$

e

$$\begin{aligned} \bar{R}(t, \cos \theta) \propto & \frac{3}{8} [|A_0|^2 + |A_{\parallel}|^2] e^{-\Gamma_s t} [Y_{0,\parallel}(t) - (1 - 2\omega)Y(t)] (1 + \cos^2 \theta) \\ & + \frac{3}{4} |A_{\perp}|^2 [Y_{\perp}(t) + (1 - 2\omega)Y(t)] (1 - \cos^2 \theta) \end{aligned} \quad (3.95)$$

Utilizzando la condizione di normalizzazione:

$$|A_0|^2 + |A_{\parallel}|^2 + |A_{\perp}|^2 = 1 \quad (3.96)$$

e ponendo:

$$R_{\perp} = \frac{|A_{\perp}|^2}{|A_0|^2 + |A_{\parallel}|^2 + |A_{\perp}|^2} \quad (3.97)$$

con:

$$1 - R_{\perp} = \frac{|A_0|^2 + |A_{\parallel}|^2}{|A_0|^2 + |A_{\parallel}|^2 + |A_{\perp}|^2} \quad (3.98)$$

otteniamo i rate di decadimento osservabili sperimentale nella forma:

$$\begin{aligned} R(t, \cos \theta) \propto & \frac{3}{8} (1 - R_{\perp}) e^{-\Gamma_s t} [Y_{0,\parallel}(t) + (1 - 2\omega)Y(t)] (1 + \cos^2 \theta) \\ & + \frac{3}{4} R_{\perp} [Y_{\perp}(t) - (1 - 2\omega)Y(t)] (1 - \cos^2 \theta) \end{aligned} \quad (3.99)$$

e:

$$\begin{aligned} \bar{R}(t, \cos \theta) \propto & \frac{3}{8} (1 - R_{\perp}) e^{-\Gamma_s t} [Y_{0,\parallel}(t) - (1 - 2\omega)Y(t)] (1 + \cos^2 \theta) \\ & + \frac{3}{4} R_{\perp} [Y_{\perp}(t) + (1 - 2\omega)Y(t)] (1 - \cos^2 \theta) \end{aligned} \quad (3.100)$$

Le due ultime espressioni possono essere scritte nella forma compatta, più conveniente per in ostri scopi, seguente:

$$\begin{aligned} R(q, t, \cos \theta) \propto & \frac{3}{8} (1 - R_{\perp}) e^{-\Gamma_s t} [Y_{0,\parallel}(t) + q(1 - 2\omega)Y(t)] (1 + \cos^2 \theta) \\ & + \frac{3}{4} R_{\perp} [Y_{\perp}(t) - q(1 - 2\omega)Y(t)] (1 - \cos^2 \theta) \end{aligned} \quad (3.101)$$

con

$$q = \begin{cases} 1 & \rightarrow B_s \\ -1 & \rightarrow \bar{B}_s \end{cases} \quad (3.102)$$

L'espressione (3.101) può essere generalizzata per tenere conto dell'efficienza di tagging ϵ_{tag} . Si può definire quindi la distribuzione di probabilità degli eventi (normalizzata all'unità), la quale per $q = 0$ definisce la distribuzione degli eventi che non è stato possibile classificare mediante la procedura di tagging:

$$f(q, t, \cos \theta) = \left[(1 - \epsilon_{tag}) (1 - q^2) + \epsilon_{tag} \frac{q^2}{2} \right] \frac{R(q, t, \cos \theta)}{\int R(q, t, \cos \theta) dq dt d\cos(\theta)} \quad (3.103)$$

3.5 Studio di sensibilità della fase di mixing del B_s^0

Per completare il lavoro di tesi ho affrontato il problema di estrarre da campioni di eventi simulati la sensibilità sulla misura della violazione della simmetria di CP , mediante l'adattamento ai dati della distribuzione (3.103), utilizzando il metodo della massima verosimiglianza.

Un evento del decadimento $B_s \rightarrow J/\psi \phi$ può essere caratterizzato in base al valore assunto dalle grandezze osservabili:

- Istante di tempo proprio di decadimento;
- Angolo di trasversalità.
- Massa invariante ricostruita;

Tralasciamo il ruolo della massa invariante, perché essa è utilizzata principalmente per la discriminazione del segnale dagli eventi di fondo. Supporremo cioè di avere ottenuto una separazione ideale del campione di segnale dal fondo. Il lavoro quindi punta ad approfondire la comprensione della metodologia utile alla determinazione dei valori degli osservabili fisici relativi alla violazione di CP , nel caso in cui lo stato finale non abbia un solo valore di CP . Risulta opportuno quindi operare una separazione dei diversi stati finali di CP su base statistica. Abbiamo visto che utilizzando l'osservabile fisico costituito dall'angolo di trasversalità è possibile scrivere un'espressione per i rate di decadimento istantanei. La simulazione consisterà quindi nella generazione di un campione di eventi nella quale la frazione degli eventi con $CP = +1$ rispetto alla frazione con $CP = -1$ è regolata dal valore assunto dal parametro R_\perp .

Inoltre per simulare la risposta del rivelatore considereremo l'effetto dell'accettanza in tempo proprio e la risoluzione strumentale nella misura del tempo proprio. Per tanto la distribuzione utilizzata per la generazione di eventi Monte Carlo è:

$$f_{obs}(q, t, \cos \theta) = \epsilon(t) \cdot [f(q, t', \cos \theta) \otimes g(t - t', \sigma_t)] \quad (3.104)$$

dove:

- $\epsilon(t)$ è la funzione di accettanza del rivelatore misurata in funzione del tempo proprio.
- $g(t - t', \sigma_t)$ è la funzione di risoluzione sperimentale di risposta dell'apparato.
- Il simbolo \otimes denota il prodotto di convoluzione delle distribuzioni di probabilità.
- $f(q, t, \cos \theta)$ è l'equazione (3.103).

Il modello di accettanza e risoluzione usato è stato ottenuto mediante il ricorso alla simulazione completa della risposta del rivelatore su dati simulati [75].

La funzione di accettanza in tempo proprio, mostrata in figura (3.3), è:

$$\epsilon(t) = \frac{(at)^3}{1 + (at)^3} \quad a = 2.8 \text{ ps}^{-1} \quad (3.105)$$

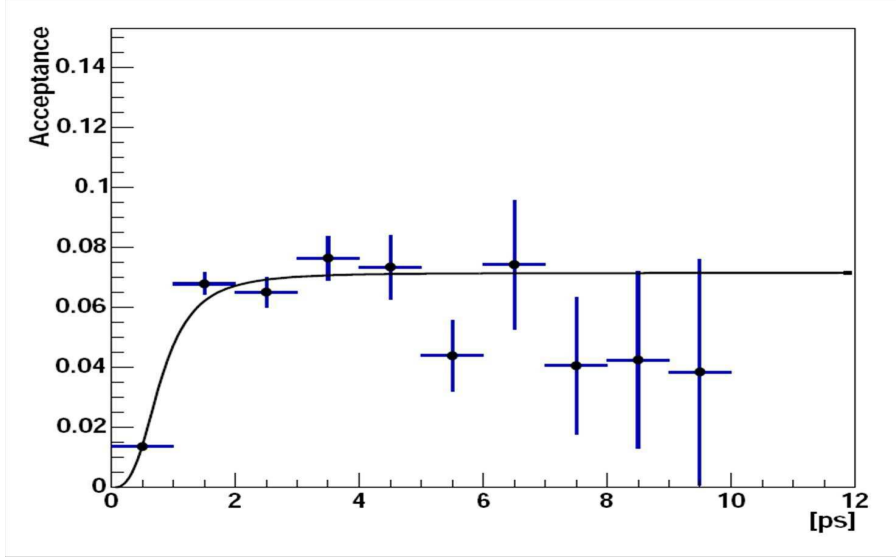


Figura 3.3: Accettanza in tempo proprio per eventi $B \leftrightarrow \bar{B}$.

La risoluzione in tempo proprio $g(t - t', \sigma_t)$ risulta essere gaussiana sull'intero spettro con una risoluzione $\sigma_t = 30$ fs.

Per semplicità sono stati trascurati gli effetti di accettanza angolare e la risoluzione angolare risultando quest'ultima dalle simulazioni Monte Carlo $\sigma(\theta) = 20$ mrad e quindi trascurabile rispetto alla variazione totale di θ nell'adattamento ai dati.

Per stimare la precisione statistica sui parametri presenti nella distribuzione (3.104) è necessario generare diversi campioni Monte Carlo simulando così diversi esperimenti. Per ciascun esperimento ho realizzato l'adattamento della distribuzione (3.104) ai dati ottenendo una stima dell'errore sui parametri. La media degli errori sui parametri ottenuti da ogni esperimento rappresenta una stima sulla precisione con cui questi parametri potranno essere misurati a LHCb. In particolare ho realizzato questa stima utilizzando la statistica che si pensa di poter selezionare in un anno di presa dati, prevista essere dell'ordine di 1.3×10^5 eventi e simulando 250 esperimenti.

La realizzazione di questo studio richiede che le coppie di valori $(t, \cos(\theta))$ siano estratti evento per evento secondo la distribuzione

La simulazione è stata realizzata utilizzando le funzionalità offerte dal codice di simulazione RooFit [76] sviluppato dalla collaborazione BaBar. Mediante RooFit è possibile definire modelli matematici per le distribuzioni di probabilità d'interesse, generare campioni di statistica prefissata e successivamente adattare distribuzioni con parametri liberi alle distribuzioni degli eventi precedentemente simulati.

I valori dei parametri utilizzati in generazione sono quelli della Tabella (3.1).

Notiamo che:

- Il valore di ΔM_s e Γ_s sono i valori centrali misurati rispettivamente in [4] [77].

$\Gamma_s [\text{ps}^{-1}]$	0.685
$\Delta\Gamma_s [\text{ps}^{-1}]$	0.1
$\Delta M_s [\text{ps}^{-1}]$	17.77
ω	0.33
ϵ_{tag}	0.56
$\phi_s [\text{rad}]$	-0.04
R_\perp	0.2

Tabella 3.1: Valori dei parametri utilizzati per la generazione di eventi simulati.

- Il valore di Γ_s è quello previsto dalla referenza [78]
- Il valore di ϕ_s è quello previsto dal Modello Standard;
- Il valore di R_\perp è stato ottenuto dalla collaborazione CDF al Fermilab [79].
- Il valore della probabilità di tagging errato ω è stato determinato dalla collaborazione LHCb mediante l'applicazione degli algoritmi di classificazione su eventi simulati, con un errore del metodo stimato di $\sigma(\omega) = 0.0036$.
- Il valore della efficienza di tagging ϵ_{tag} è stato determinato dalla collaborazione LHCb mediante l'applicazione degli algoritmi di classificazione su eventi simulati.

Sottolineiamo che la probabilità di tagging errato deve essere determinata utilizzando canali di decadimento di calibrazione: nel caso preso in considerazione non può essere determinata utilizzando solo l'adattamento delle distribuzioni agli eventi corrispondenti al canale in esame.

Un esempio della distribuzione dei tempi propri di decadimento senza includere la funzione di accettazione e la risoluzione in tempo proprio è mostrata in figura (3.4) in riferimento alla statistica di un anno di presa dati. Un esempio di distribuzione dell'osservabile $\cos \theta$ è mostrata in figura (3.5) in riferimento alla statistica di un anno di presa dati.

L'adattamento ai dati è stato realizzato supponendo di aver misurato la probabilità di tagging errato $\bar{\omega} \pm \sigma(\omega)$ e $\overline{\Delta M_s} \pm \sigma(\Delta M_s)$ dal decadimento $B_s \rightarrow D_s \pi$ ad LHCb [75]. Per tanto la distribuzione finale adattata ai dati risulta essere:

$$\begin{aligned} \mathcal{F}_{obs}(q, t, \cos \theta) = & \epsilon(t) \cdot [f(q, t', \cos \theta) \otimes g(t - t', \sigma_t)] \cdot \\ & \cdot \exp\left(\frac{(\Delta M_s - \overline{\Delta M_s})^2}{2 \cdot \sigma^2(\Delta M_s)}\right) \cdot \exp\left(\frac{(\omega - \bar{\omega})^2}{2 \cdot \sigma^2(\omega)}\right) \end{aligned} \quad (3.106)$$

In tabella (3.2) sono riportate le medie delle stime degli errori corrispondenti a 250 esperimenti. Nelle figure (3.6), (3.7), (3.8), (3.9), (3.10) sono riportate le distribuzioni dei valori centrali e degli errori ottenuti dall'adattamento ai dati per ciascun esperimento, così come le distribuzioni di $pull^1$ che risultano essere tutte canoniche².

¹La distribuzione $pull$ è definita come la differenza fra il valore usato in generazione e il valore ottenuto dall'adattamento ai dati diviso l'errore ottenuto dall'adattamento ai dati del

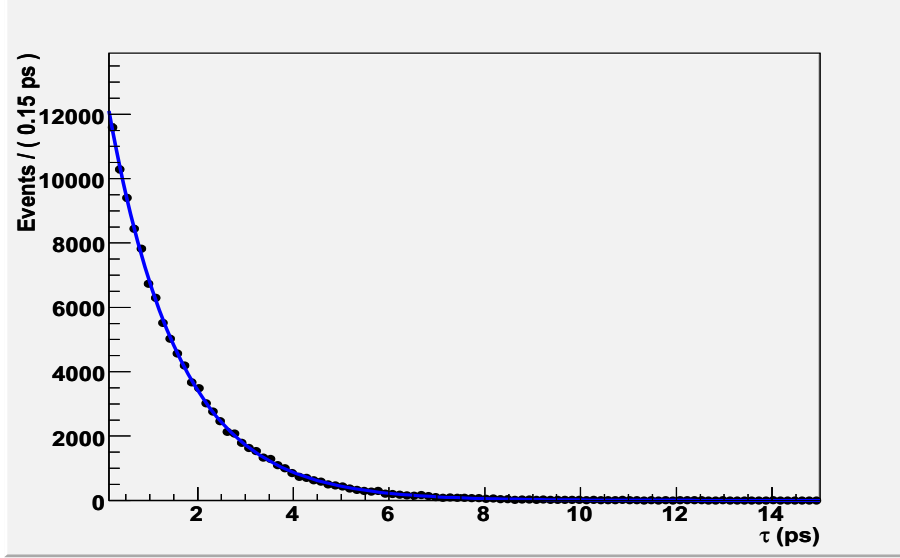


Figura 3.4: Distribuzione del tempo proprio per eventi generati B, \bar{B} con sovrapposta la funzione (3.103) integrata in $\cos\theta$ e q .

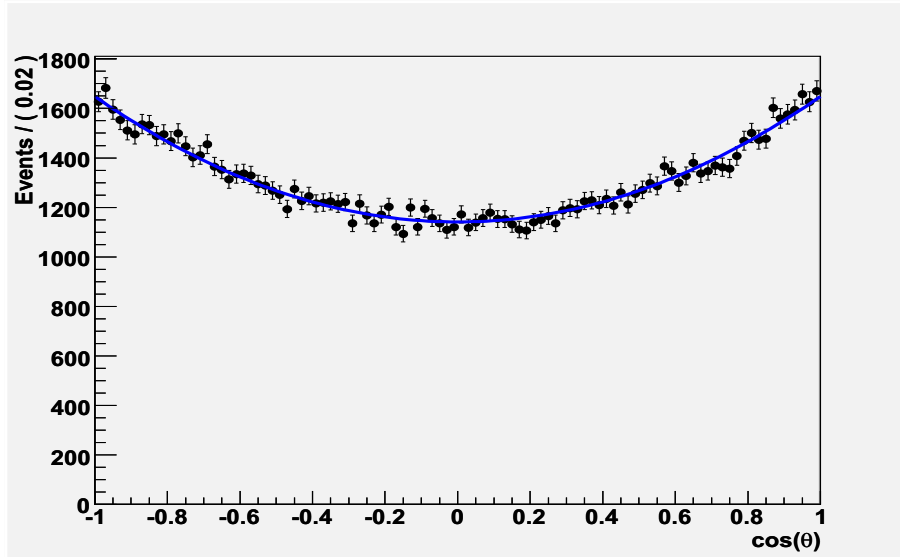
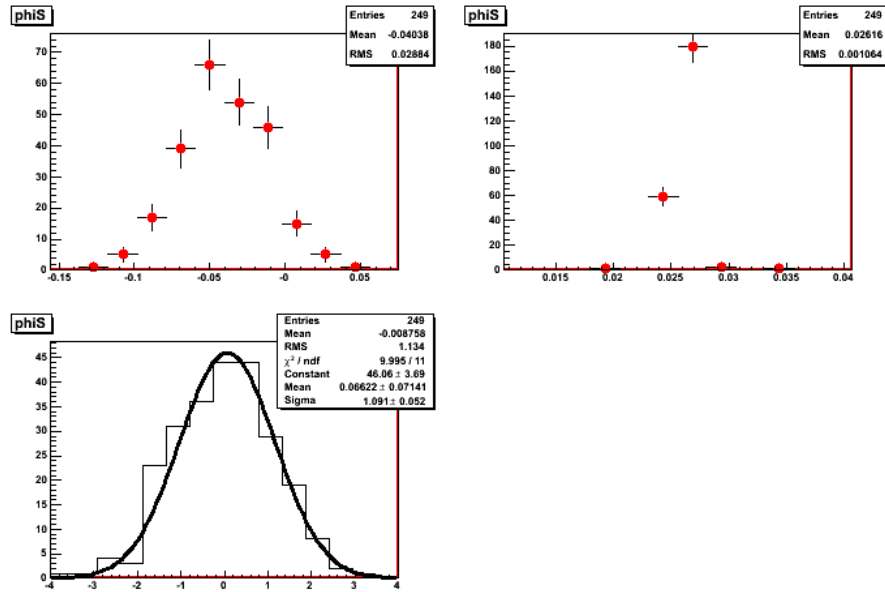


Figura 3.5: Distribuzione di $\cos\theta$ per eventi generati B, \bar{B} con sovrapposta la funzione (3.103) integrata nel tempo proprio e in q .

Parametri	Media degli errori (250 esperimenti)
R_{\perp}	0.0057
$\phi_s [\text{rad}]$	0.026
$\Delta\Gamma_s [\text{ps}^{-1}]$	0.013
$\Delta M_s [\text{ps}^{-1}]$	0.007
ω	0.0036
$\Gamma_s [\text{ps}^{-1}]$	0.0034

Tabella 3.2: Stima della sensibilità dei parametri della distribuzione (3.106).

Figura 3.6: In alto: distribuzione dei risultati ottenuti dall'adattamento ai dati dei 250 esperimenti per il parametro ϕ_s , valore centrale (a sinistra), errore (a destra). In basso: *pull* del parametro ϕ_s .

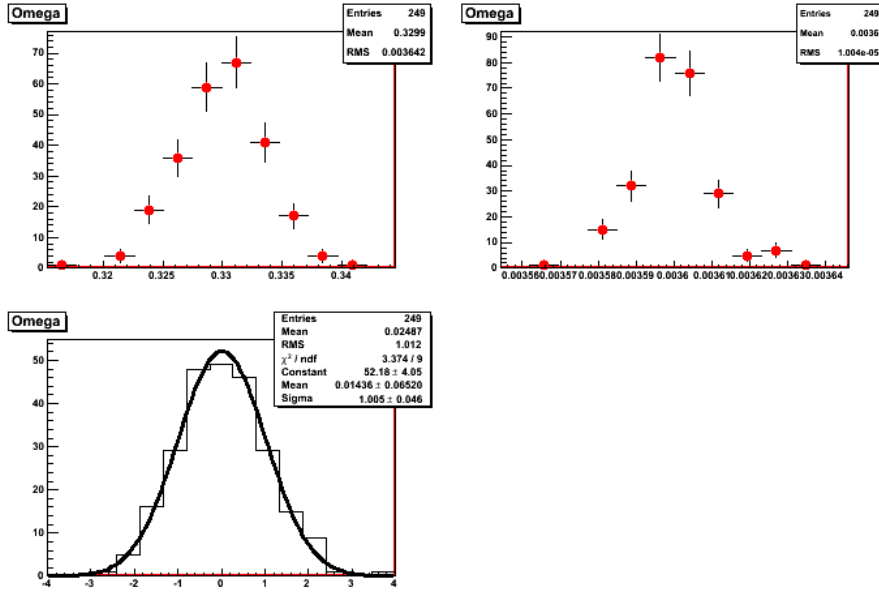


Figura 3.7: In alto: distribuzione dei risultati ottenuti dall'adattamento ai dati dei 250 esperimenti per il parametro ω , valore centrale (a sinistra), errore (a destra). In basso: *pull* del parametro ω .

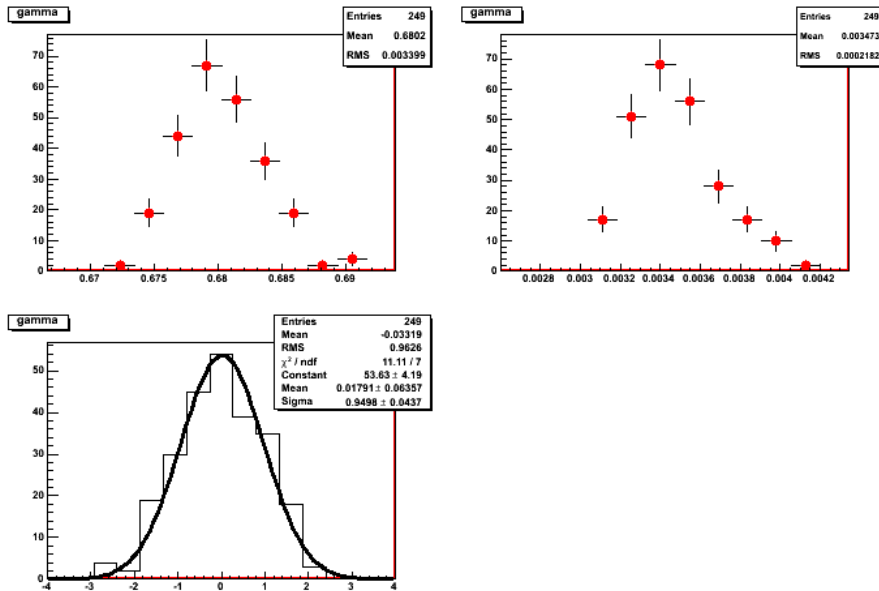


Figura 3.8: In alto: distribuzione dei risultati ottenuti dall'adattamento ai dati dei 250 esperimenti per il parametro Γ , valore centrale (a sinistra), errore (a destra). In basso: *pull* del parametro Γ .

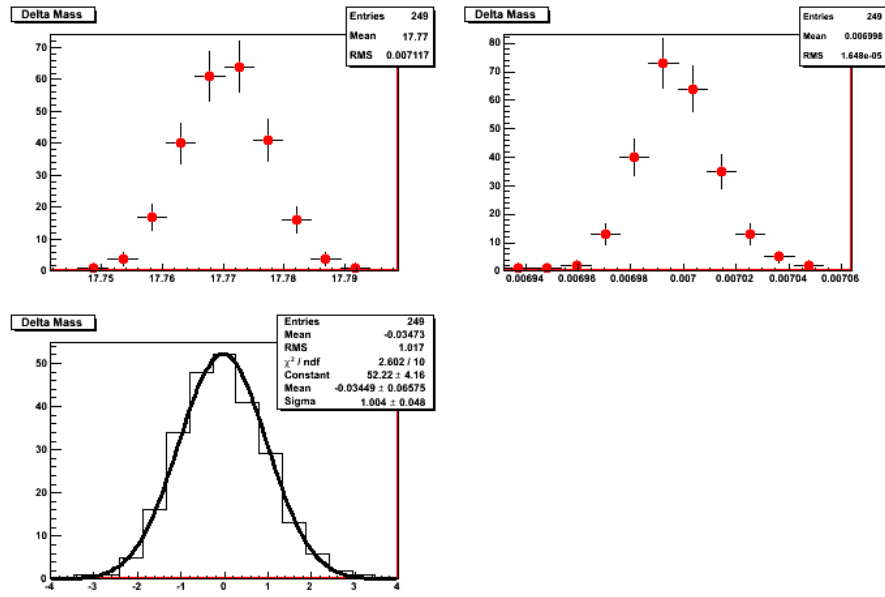


Figura 3.9: In alto: distribuzione dei risultati ottenuti dall'adattamento ai dati dei 250 esperimenti per il parametro ΔM_s , valore centrale (a sinistra), errore (a destra). In basso: *pull* del parametro ΔM_s .

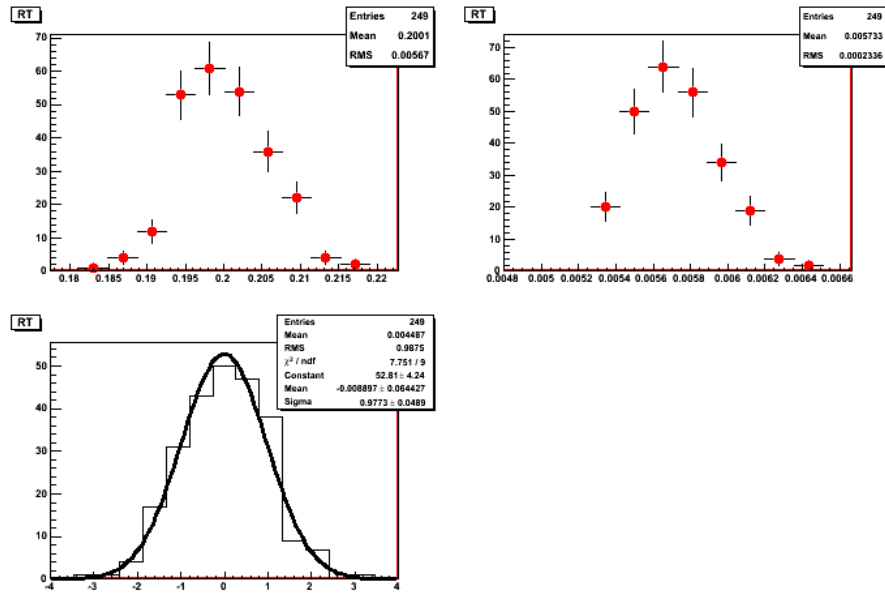


Figura 3.10: In alto: distribuzione dei risultati ottenuti dall'adattamento ai dati dei 250 esperimenti per il parametro R_{\perp} , valore centrale (a sinistra), errore (a destra). In basso: *pull* del parametro R_{\perp} .

3.6 Conclusioni

Lo studio realizzato in questa tesi, descritto in questo capitolo, mostra che operando in condizioni pressoché ideali sarebbe possibile raggiungere una precisione statistica nella determinazione della fase di miscelamento dei mesoni B_s^0 , in un solo anno di presa dati, davvero considerevole: $\sigma(\phi_s) \sim 0.03$.

Grazie a questa tecnica di misura sarebbero dunque rivelabili effetti anche minimi che si discostassero dalle previsioni del Modello Standard a condizione di accumulare eventi per qualche anno di presa di dati.

Naturalmente le condizioni reali rendono la misura decisamente più complessa e meno precisa. Ci sono diversi effetti di cui tenere conto, che concorrono a complicare il quadro sperimentale. Certamente l'ipotesi di eliminare completamente il fondo costituisce una semplificazione e inoltre non abbiamo tenuto conto degli effetti della risoluzione e dell'accettanza angolare sulla precisione statistica.

D'altra parte però si può osservare che l'analisi angolare può essere sviluppata ulteriormente, in modo da elevare il potere discriminante del metodo di misura sia nel riconoscimento degli stati di diverso valore di CP sia nel riconoscimento del fondo, ricorrendo alla descrizione completa del decadimento in tre angoli, anziché limitare la trattazione al solo angolo di discriminazione θ .

parametro libero.

²Una distribuzione è canonica se ha deviazione standard unitaria e media zero, che indica che i valori adattati e i loro errori sono stimati correttamente.

Conclusioni

Il lavoro svolto durante il dottorato di ricerca, che ho presentato in questa tesi, ha riguardato due attività ben distinte. Dapprima mi sono occupato della realizzazione del sistema di controllo e configurazione della farm di trigger di alto livello dell'esperimento LHCb; successivamente ho lavorato allo studio di fattibilità della misura della fase di miscelamento del mesone B_s , dovuta all'oscillazione, mediante la rivelazione del decadimento $B_s \rightarrow J/\psi\phi$ con la tecnica dell'analisi angolare.

Il lavoro sviluppato per realizzare il sistema di controllo e configurazione della farm si è rivelato molto impegnativo e ha richiesto più di due anni per essere impostato. Il prodotto finale, in fase di collaudo oggi, ha ricevuto importanti riconoscimenti da parte della collaborazione LHCb ed è stato adottato dal CERN come progetto generale a disposizione degli esperimenti che ne facciano richiesta (al momento alcune sue componenti sono utilizzate dagli esperimenti ATLAS e CMS).

Il lavoro mi ha permesso di raggiungere una certa competenza nel controllo di farm di computer, equipaggiati con sistemi operativi Linux e organizzati in reti LAN, e di maturare anche esperienza nella definizione delle configurazioni ottimali dei sistemi operativi.

Recentemente mi è stato possibile mettere a frutto l'esperienza maturata grazie ad un contratto di collaborazione che ho ottenuto presso il centro di calcolo nazionale dell'INFN presso il CNAF di Bologna, grazie al quale avrò modo di proseguire la mia attività di ricerca nell'INFN, occupandomi dei problemi connessi alla gestione e funzionamento di un centro di grande complessità.

Ho voluto affrontare poi il problema della determinazione di uno degli osservabili più importanti per la verifica delle previsioni del Modello Standard, osservabile fra i più sensibili, secondo vari modelli teorici, a possibili contributi di nuova fisica.

Il decadimento ha una segnatura sperimentale (la presenza di quattro particelle cariche nello stato finale prodotte dai decadimenti delle particelle $J/\psi \rightarrow \mu^+\mu^-$ e $\phi \rightarrow K^+K^-$) che lo rendono particolarmente ben identificabile nel contesto dell'esperimento LHCb, con contaminazioni dovute ad eventi di fondo ritenute in base alle simulazioni esigue.

Per contro il decadimento del mesone B_s in questo processo di decadimento conduce ad uno stato finale che può assumere autovalori di CP diversi (+1 e -1) che debbono essere sperati pena altrimenti la diluzione del valore degli osservabili di interesse dovuta alla sovrapposizione di effetti di segno opposto. Ho mostrato che è possibile realizzare una descrizione del decadimento mediante

variabili angolari opportune, utilizzando le quali è possibile separare su base statistica i contributi allo stato finale corrispondenti ai due diversi stati di CP possibili. Ho quindi determinato le espressioni delle distribuzioni di probabilità di decadimento dei mesoni B_s e \bar{B}_s in funzione del tempo e proprio e di un angolo (detto di trasversalità) che permette di separare le componenti di CP diverso. Per mezzo del codice di simulazione RooFit, sviluppato dalla collaborazione BaBar, ho quindi realizzato la simulazione di misure ripetute con metodo Monte Carlo. Ciascuna misura produce un campione di taglia corrispondente alla durata di un anno di presa dati di LHCb. Ho quindi utilizzato il metodo della massima verosimiglianza *unbinned* per adattare le distribuzioni ai dati simulati e determinare l'ordine di grandezza dell'errore statistico di misura della fase di miscelamento. Il risultato è particolarmente incoraggiante perché pur non sfruttando appieno il potere di separazione derivante dall'impiego delle informazioni relative alle distribuzioni angolari dei prodotti di decadimento indica la possibilità di raggiungere precisioni statistiche di alcuni centesimi di radiante in un solo anno di presa dati.

Due delle ipotesi in base alle quali ho sviluppato il modello sono particolarmente ottimistiche: la prima è quella relativa all'effetto del fondo, che come detto nel modello è ritenuto trascurabile; la seconda riguarda la precisione di misura delle coordinate angolari, ritenuta ideale perché affetta da errori trascurabili, e senza perdita di eventi a causa di tagli in accettazione.

Il mio contributo in questo ambito è certamente preliminare, e meriterebbe sviluppi e approfondimenti ulteriori. Non di meno, è stata un'attività di estremo interesse personale, che sarà proseguita dal gruppo INFN di LHCb della Sezione di Bologna presso il quale ho avuto il piacere di lavorare in pieno spirito di collaborazione.

Ringraziamenti e pensieri finali

Ringraziamenti, finalmente è arrivata la pagina dei ringraziamenti di questo lungo cammino iniziato quattro anni fa. Per primo voglio ringraziare Angelo, Daniela, Domenico, Gianluca, Vincenzo e Umberto per la pazienza e le cose che mi hanno insegnato soprattutto al di fuori della fisica. Si meriterebbero una medaglia per essere riusciti a sopportarmi per tutto questo tempo. Senza di loro questo traguardo non sarebbe stato raggiunto.

Guardando indietro vedo un percorso iniziato perdendo delle persone che amavo. Ho riscoperto da subito tanti amici veri che mi sono stati vicini, alcuni che conoscevo da tanti anni, altri li ho incontrati lungo questa strada, altri come sono apparsi altrettanto velocemente si sono allontanati ma hanno lasciato comunque un ricordo. Con loro sono stato e sto bene, mi sento di ringraziarli.

Nessuno conosce il proprio destino e forse la cosa più difficile è cercare di affrontarlo.

Bibliografia

- [1] *PYTHIA home page*. [Online]. Available: <http://www.thep.lu.se/~torbjorn/Pythia.html>
- [2] LHCb collaboration, *LHCb reoptimized detector TDR*, CERN-LHCC/2003-030, ISBN 92-9083-209-6. [Online]. Available: <http://lhcb.web.cern.ch/lhcb/TDR/TDR9.pdf>
- [3] LHCb collaboration, *LHCb VELO TDR*, CERN-LHCC/2001-028, ISBN 92-9083-179-X. [Online]. Available: http://lhcb-vd.web.cern.ch/lhcb-vd/TDR/velo_tdr.pdf
- [4] Particle Data Group, *The Review of Particle Physics*. [Online]. Available: <http://pdg.lbl.gov>
- [5] N. Zaitsev, *The LHCb Vertex Trigger*, Presentation at Vertex99. [Online]. Available: <http://lhcb-doc.web.cern.ch/lhcb-doc/presentations/conferencetalks/postscript/1999presentations/vertex99zaitsev.ps>
- [6] N. Zaitsev, *Study of the LHCb pile-up trigger and $B_s \rightarrow J/\Psi\Phi$ decay*, Ph.D. Thesis, Amsterdam Univ., October 2000.
- [7] F. Legger, *Contribution to the development of the LHCb acquisition electronics and study of polarized radiative Λ_b decays*, Ph.D. Thesis, Lausanne Univ., 2006.
- [8] LHCb collaboration, *LHCb inner tracker TDR*, CERN-LHCC/2003-029, ISBN 92-9083-201-X. [Online]. Available: <http://lhcb.web.cern.ch/lhcb/TDR/front/%20cover/LHCb-IT-TDR.pdf>
- [9] Leonar Bart Anton Hommels, *The tracker in the trigger of LHCb*, Ph.D. Thesis, october 2006.
- [10] LHCb collaboration, *LHCb outer tracker TDR*, CERN-LHCC-2001-041, ISBN 92-9083-200-2. [Online]. Available: <http://www.nikhef.nl/pub/experiments/bfys/lhcb/outerTracker/tdr/final.pdf>
- [11] A. G. Bates, *Developments in silicon detectors and their impact in LHCb physics measurements*, Ph.D. Thesis, Glasgow Univ., UK, September 2005.

- [12] LHCb collaboration, *LHCb calorimeters TDR*, CERN-LHCC-2000-049, ISBN 92-9083-169-3. [Online]. Available: http://lhcb.cern.ch/calorimeters/html/TDR/calor_tdr.pdf
- [13] LHCb collaboration, *LHCb muon system TDR*, CERN-LHCC-2001-010, and addendum: CERN-LHCC-2003-002, CERN-LHCC-2005-0012, ISBN 92-9083-180-4. [Online]. Available: <http://lhcb-muon.web.cern.ch/lhcb-muon/results/tdr.pdf>
- [14] LHCb collaboration, *LHCb trigger system TDR*, CERN-LHCC-2003-031, ISBN 92-9083-208-8. [Online]. Available: <http://lhcb.web.cern.ch/lhcb/TDR/trigtdr.pdf>
- [15] A. Barczyk, G. Haefeli, R. Jacobsson, B. Jost, and N. Neufeld., *1 MHz Readout*, CERN, LHCb note 2005-062 DAQ, Sept. 7, 2005. [Online]. Available: <http://doc.cern.ch/archive/electronic/cern/others/LHB/public/lhcb-2005-062.pdf>.
- [16] L. Fernandez, P. Koppenburg, *Esclusive HLT performance*, LHCb note 2005-047. [Online]. Available: <http://doc.cern.ch/archive/electronic/cern/others/LHB/public/lhcb-2005-047.pdf>
- [17] PVSS-II, home page. [Online]. Available: <http://www.pvss.com>
- [18] JCOP, home page. [Online]. Available: <http://itco.web.cern.ch/itco/Projects-Services/JCOP>
- [19] IEEE Standards Coordinating Committee 14, *IEEE Trial-Use Standard for Prefixes for Binary Multiples*, The Institute of Electrical and Electronics Engineers. [Online]. Available: <http://ieeexplore.ieee.org/iel5/8450/26611/01186538.pdf>
- [20] C. Gaspar, *SMI++ A Finite State Machine Toolkit* http://itco.web.cern.ch/itco/Projects-Services/JCOP/ProjectTeam/Minutes/1999/23-06-99/smi_jcop/index.htm
- [21] Artur Barczyk, Daniela Bortolotti, Angelo Carbone, Jean-Pierre Dufey, Domenico Galli, Benjamin Gaidioz, Daniele Gregori, Beat Jost, Umberto Marconi, Niko Neufeld, Gianluca Peco, and Vincenzo Maria Vagnoni, *High rate packets transmission on Ethernet LAN using commodity hardware*, IEEE Trans. Nucl. Sci., vol. 53, pp. 810-816, 2006. [Online]. Available: <http://ieeexplore.ieee.org/iel/23/34476/012644946.pdf>
- [22] CERN IT Division: *Farm Monitoring and Control*. <http://itcobe.web.cern.ch/itcobe/Projects/Framework/Download/Components/SystemOverview/fwFMC/welcome.html>
- [23] C. Gaspar, D. Donszelmann, *DIM - A Distributed Information Management System for the DELPHI experiment at CERN*, <http://dim.web.cern.ch/dim/papers/DIM.PS>

- [24] Federico Bonifazi, Daniela Bortolotti, Angelo Carbone, Domenico Galli, Daniele Gregori, Umberto Marconi, Gianluca Peco, Vincenzo Maria Vagnoni, *The Message Logger for the LHCb on-line farm*, CERN, LHCb note 2005-050 DAQ, 23 Agosto 2005. [Online]. Available: <http://doc.cern.ch/archive/electronic/cern/others/LHB/public/lhcb-2005-050.pdf>.
- [25] `fifo(4)` - Linux man page. [Online]. Available: <http://www.die.net/doc/linux/man/man4/fifo.4.html>
- [26] `pipe(7)` - Linux man page. [Online]. Available: <http://www.die.net/doc/linux/man/man7/pipe.7.html>
- [27] M. Allman, V. Paxson, W. Stevens, *TCP Congestion Control*, RFC 2581. [Online]. Available: <http://www.ietf.org/rfc/rfc2581.txt>
- [28] Circular pipes. [Online]. Available: <http://lwn.net/Articles/118750/>
- [29] `signal(7)` - Linux man page. [Online]. Available: <http://www.die.net/doc/linux/man/man7/signal.7.html>
- [30] `fcntl(2)` - Linux man page. [Online]. Available: <http://www.die.net/doc/linux/man/man2/fcntl.2.html>
- [31] `sigtimedwait(2)` - Linux man page. [Online]. Available: <http://www.die.net/doc/linux/man/man2/sigtimedwait.2.html>
- [32] `glob(7)` - Linux man page. [Online]. Available: <http://www.die.net/doc/linux/man/man7/glob.7.html>
- [33] `regex(7)` - Linux man page. [Online]. Available: <http://www.die.net/doc/linux/man/man7/regex.7.html>
- [34] `read(2)` - Linux man page. [Online]. Available: <http://www.die.net/doc/linux/man/man2/read.2.html>
- [35] V.I. Levenshtein, *Binary codes capable of correcting deletions, insertions and reversals*, Soviet physics Doklady, vol. 10, nr. 8, pp. 707-710, 1966.
- [36] Federico Bonifazi, Daniela Bortolotti, Angelo Carbone, Domenico Galli, Daniele Gregori, Umberto Marconi, Gianluca Peco, Vincenzo Maria Vagnoni, *The Power Manager for the LHCb on-line farm*, CERN, LHCb note 2007-094 DAQ, 9 Luglio 2007. [Online]. Available: <http://doc.cern.ch/archive/electronic/cern/others/LHB/public/lhcb-2007-094.pdf>.
- [37] *Intelligent Management Platform Interface*. [Online]. Available: <http://www.intel.com/design/servers/ipmi/>
- [38] *OpenIPMI*. [Online]. Available: <http://openipmi.sourceforge.net/>

- [39] R. Droms, *Dinamical Host Configuration Protocol*, RFC 2131. [Online]. Available: <http://tools.ietf.org/html/rfc2131>
- [40] Federico Bonifazi, Daniela Bortolotti, Angelo Carbone, Domenico Galli, Daniele Gregori, Umberto Marconi, Gianluca Pecco, Vincenzo Maria Vagnoni, *The Task Manager for the LHCb on-line farm*, CERN, LHCb note 2004-099 DAQ, 24 Novembre 2004. [Online]. Available: <http://doc.cern.ch//archive/electronic/cern/others/LHB/public/lhcb-2004-099.pdf>
- [41] `environ(5)` - Linux man page. [Online]. Available: <http://www.die.net/doc/linux/man/man5/environ.5.html>
- [42] T.Bowden, B.Bauer, J.Narin, *The /proc file system*. [Online]. Available: <http://lxr.linux.no/source/Documentation/filesystems/proc.txt>
- [43] `execve(2)` - Linux man page. [Online]. Available: <http://www.die.net/doc/linux/man/man2/execve.2.html>
- [44] `fork(2)` - Linux man page. [Online]. Available: <http://www.die.net/doc/linux/man/man2/fork.2.html>
- [45] `chdir(2)` - Linux man page. [Online]. Available: <http://www.die.net/doc/linux/man/man2/chdir.2.html>
- [46] `clearenv(3)` - Linux man page. [Online]. Available: <http://www.die.net/doc/linux/man/man3/clearenv.3.html>
- [47] `putenv(3)` - Linux man page. [Online]. Available: <http://www.die.net/doc/linux/man/man3/putenv.3.html>
- [48] `setenv(3)` - Linux man page. [Online]. Available: <http://www.die.net/doc/linux/man/man3/setenv.3.html>
- [49] `close(2)` - Linux man page. [Online]. Available: <http://www.die.net/doc/linux/man/man2/close.2.html>
- [50] `open(2)` - Linux man page. [Online]. Available: <http://www.die.net/doc/linux/man/man2/open.2.html>
- [51] `dup2(2)` - Linux man page. [Online]. Available: <http://www.die.net/doc/linux/man/man2/dup2.2.html>
- [52] `sched_setaffinity(2)` - Linux man page. [Online]. Available: http://www.die.net/doc/linux/man/man2/sched_setaffinity.2.html
- [53] `sched_setscheduler(2)` - Linux man page. [Online]. Available: http://www.die.net/doc/linux/man/man2/sched_setscheduler.2.html
- [54] `setpriority(2)` - Linux man page. [Online]. Available: <http://www.die.net/doc/linux/man/man2/setpriority.2.html>

- [55] `setuid(2)` - Linux man page. [Online]. Available: <http://www.die.net/doc/linux/man/man2/setuid.2.html>
- [56] `setgid(2)` - Linux man page. [Online]. Available: <http://www.die.net/doc/linux/man/man2/setgid.2.html>
- [57] `umask(2)` - Linux man page. [Online]. Available: <http://www.die.net/doc/linux/man/man2/umask.2.html>
- [58] `setsid(2)` - Linux man page. [Online]. Available: <http://www.die.net/doc/linux/man/man2/setsid.2.html>
- [59] `kill(2)` - Linux man page. [Online]. Available: <http://www.die.net/doc/linux/man/man2/kill.2.html>
- [60] `pthread_sigmask(2)` - Linux man page [Online]. Available: http://www.die.net/doc/linux/man/man2/pthread_sigmask.2.html
- [61] `waitpid(2)` - Linux man page. [Online]. Available: <http://www.die.net/doc/linux/man/man2/waitpid.2.html>
- [62] `_exit(2)` - Linux man page. [Online]. Available: http://www.die.net/doc/linux/man/man2/_exit.2.html
- [63] `exit(3)` - Linux man page. [Online]. Available: <http://www.die.net/doc/linux/man/man3/exit.3.html>
- [64] Federico Bonifazi, Daniela Bortolotti, Angelo Carbone, Domenico Galli, Daniele Gregori, Umberto Marconi, Gianluca Peco, Vincenzo Maria Vagnoni, *The Process Controller for the LHCb on-line farm*, CERN, LHCb note 2005-095 DAQ, 9 luglio 2007. [Online]. Available: <http://doc.cern.ch/archive/electronic/cern/others/LHB/public/lhcb-2007-095.pdf>
- [65] Federico Bonifazi, Daniela Bortolotti, Angelo Carbone, Domenico Galli, Daniele Gregori, Umberto Marconi, Gianluca Peco, Vincenzo Maria Vagnoni, *The Monitor System for the LHCb on-line farm*, CERN, LHCb note 2005-051 DAQ, 31 Agosto 2005. [Online]. Available: <http://doc.cern.ch/archive/electronic/cern/others/LHB/public/lhcb-2005-051.pdf>
- [66] `brk(2)` - Linux man page. [Online]. Available: <http://www.die.net/doc/linux/man/man2/brk.2.html>
- [67] P.Mochel, *The sysfs Filesystem*. [Online]. Available: <http://www.kernel.org/pub/linux/kernel/people/mochel/doc/papers/ols-2005/mochel.pdf>
- [68] *Sysfs, the filesystem for exporting kernel objects*. [Online]. Available: <http://delcom/sourceforge.net/index>
- [69] *The /proc file system utilities*. [Online]. Available: <http://procp.sourceforge.net/index.html>

- [70] BogoMips Mini-Howto [Online]. Available: <http://www.clifton.nl/index.html?bogomips.html>
- [71] *Small Form Factor Committee Specification Draft for S.M.A.R.T. Application Guide for the ATA and SCSI interfaces SFF8055i*. [Online]. Available: <ftp://ftp3.ds.pg.gda.pl/people/macro/S.M.A.R.T./8055.PDF>
- [72] *SFF Committee Specification for Self-Monitoring, Analysis and Reporting Technology (S.M.A.R.T.) SFF-8035i Revision 2.0*. [Online]. Available: ftp://ftp3.ds.pg.gda.pl/people/macro/S.M.A.R.T./8035R2_0.PDF
- [73] Weisskopf V. and Wigner E., Z. Phys., 63 54 (1930)
- [74] G. Buchalla, A. J. Buras and Markus E. Lautenbacher, Rev. Mod. Phys. 68, 1125 (1996).
- [75] L. Fernandez, *Exclusive Trigger Selections and Sensitivity to the $B_s - \bar{B}_s$ Mixing Phase at LHCb* Ph.D. Thesis 2006. [Online]. Available: <http://documents.cern.ch/cgi-bin/setlink?base=preprint&categ=cern&id=cern-thesis-2006-042>
- [76] RooFit home page, *The RooFit toolkit for data modelling*. [Online]. Available: <http://roofit.sourceforge.net/>
- [77] A. Abulencia et al., *Observation of $B_s^0 - \bar{B}_s^0$ oscillations*, Phys. Rev. Lett., 97:242003, hep-ex/0609040, 2006.
- [78] Ball P., Fleischer R., *Probing new physics through B mixing: Status, benchmark and prospects*, Eur. Phys. J. C48:413-426, hep-ph/0604249v1, 2006.
- [79] D. Acosta et al., *Measurement of the timelife difference between B_s mass eigenstates*, Phys. Rev. Lett., 94:101803, 2005.